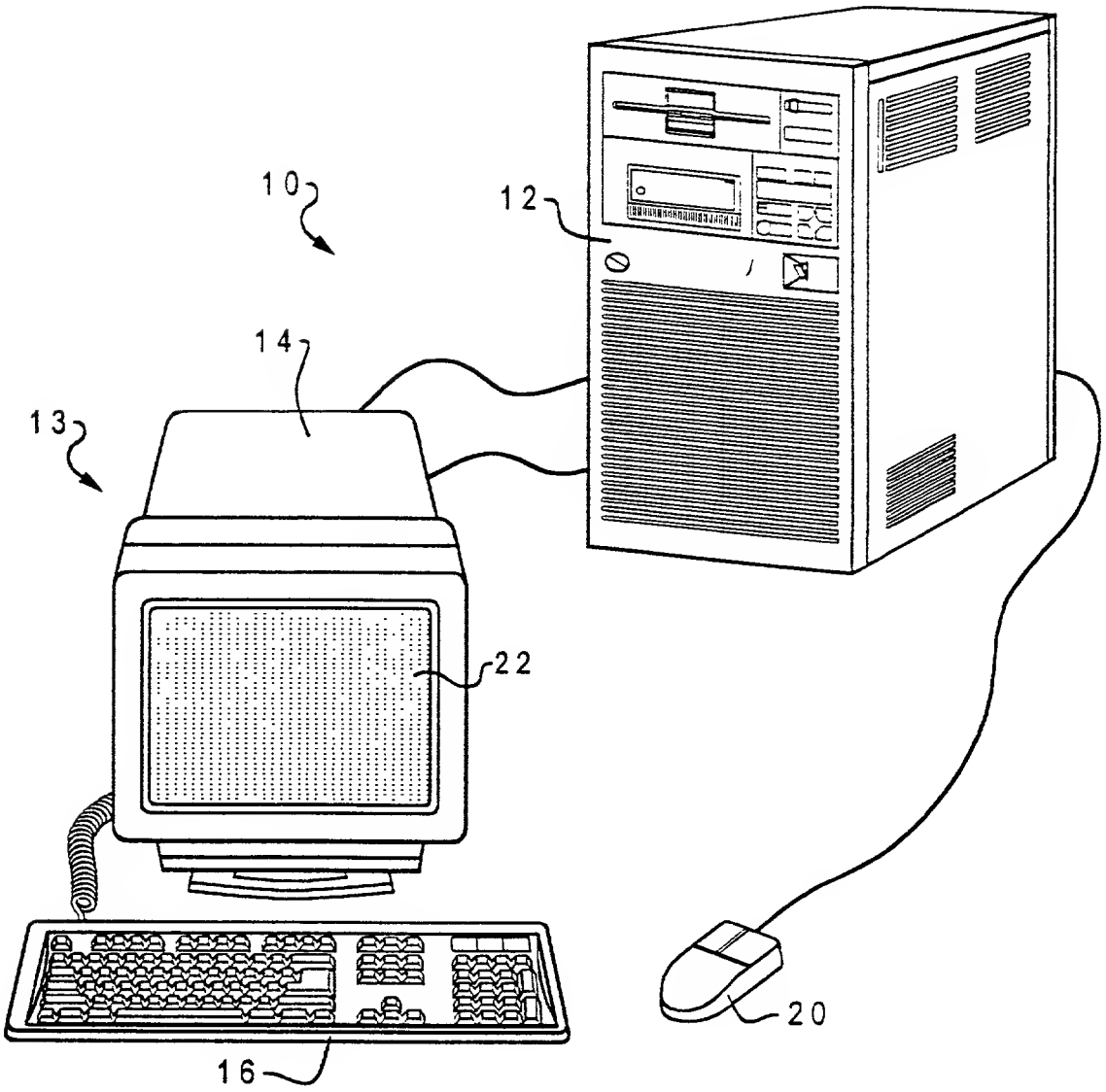


Fig. 1

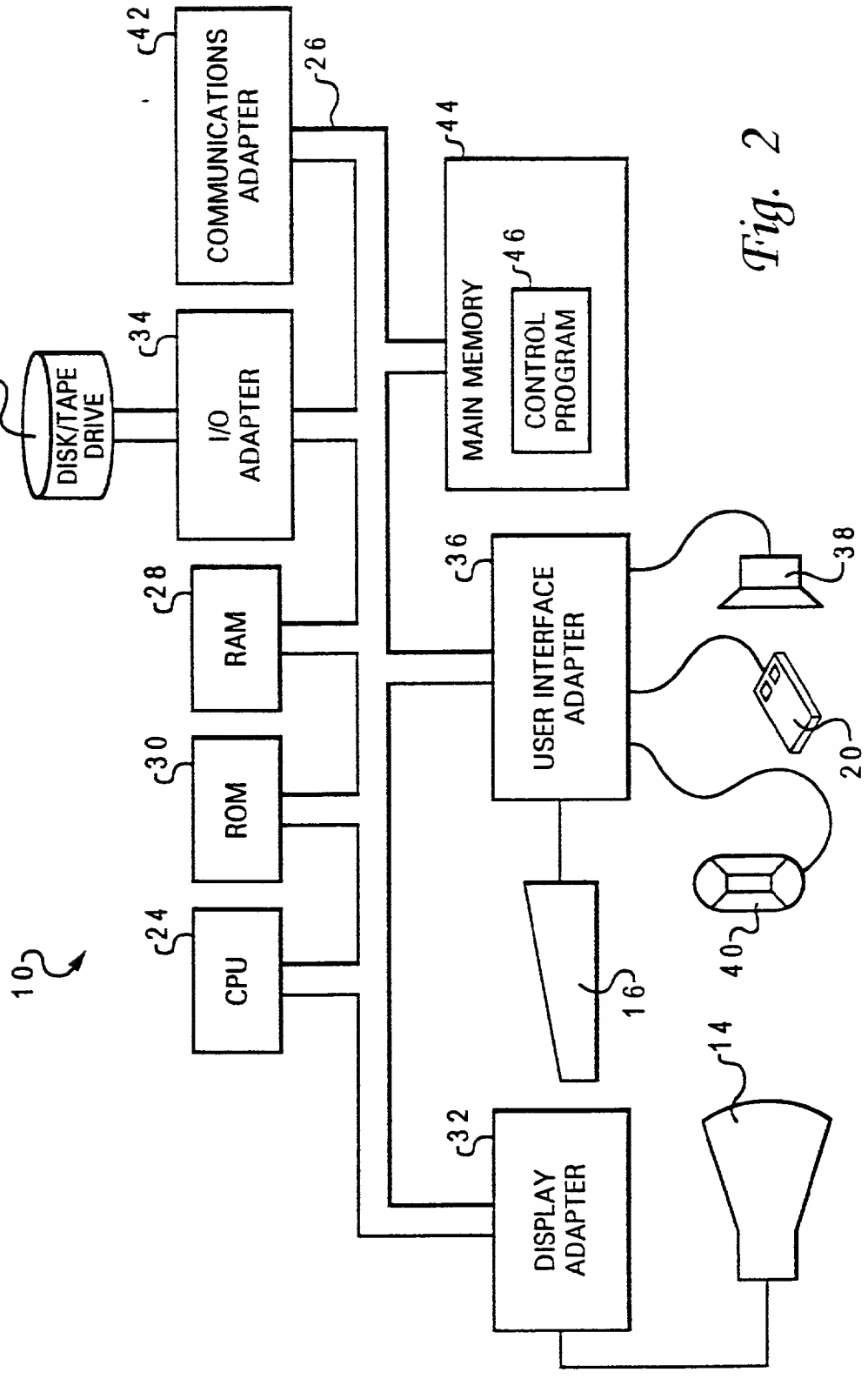

10
CPU
24
ROM
30
RAM
28
I/O ADAPTER
34
DISK/TAPE DRIVE
33
COMMUNICATIONS ADAPTER
42
26
MAIN MEMORY
44
CONTROL PROGRAM
46
USER INTERFACE ADAPTER
36
DISPLAY ADAPTER
32
16
14
40
20
38

Fig. 2

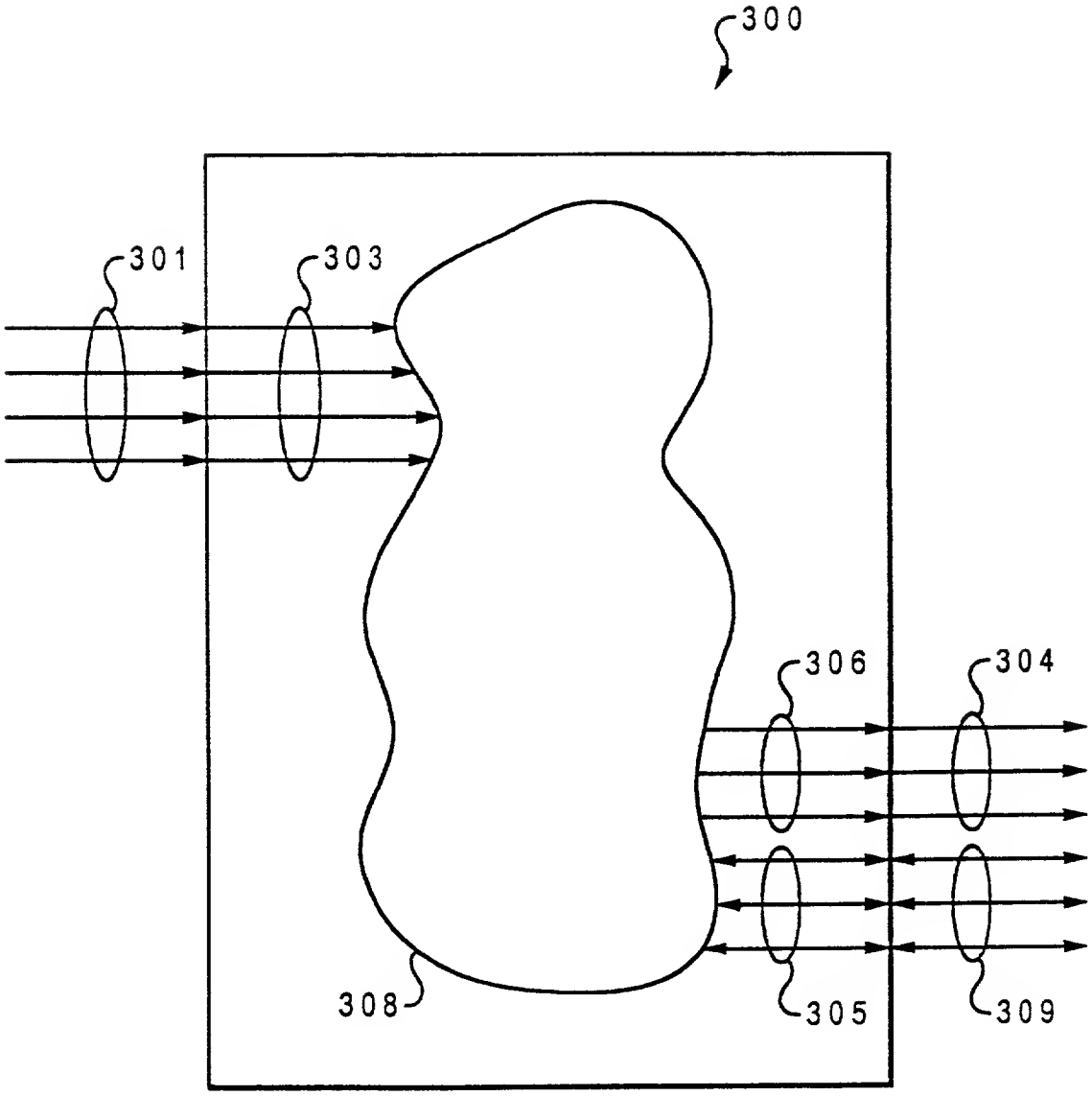


Fig. 3A

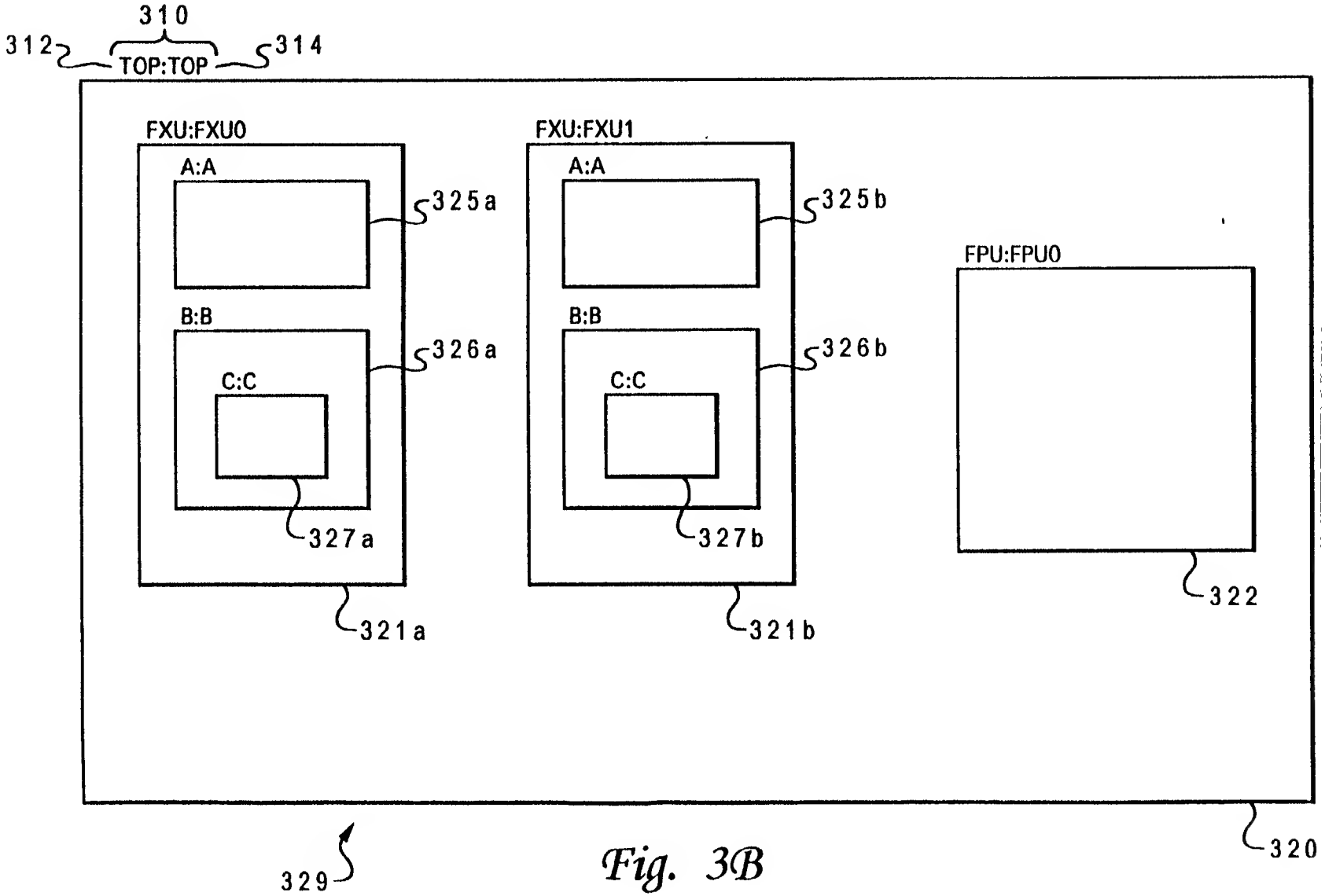

310
312
TOP:TOP
314
FXU:FXU0
A:A
325a
B:B
326a
C:C
327a
321a
FXU:FXU1
A:A
325b
B:B
326b
C:C
327b
321b
FPU:FPU0
322
320
329

Fig. 3B

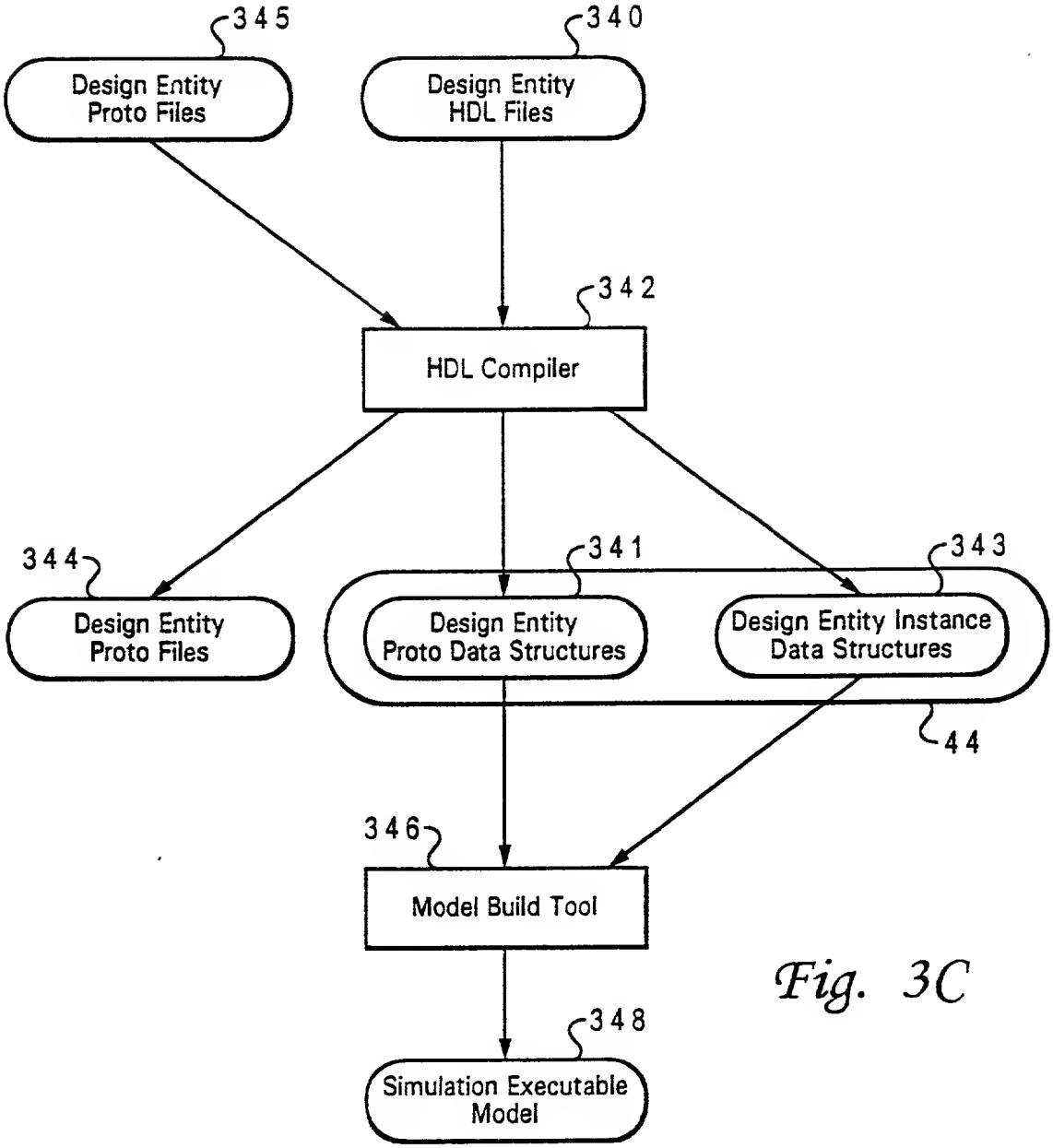

345
Design Entity Proto Files
340
Design Entity HDL Files
342
HDL Compiler
344
Design Entity Proto Files
341
Design Entity Proto Data Structures
343
Design Entity Instance Data Structures
44
346
Model Build Tool
348
Simulation Executable Model

Fig. 3C

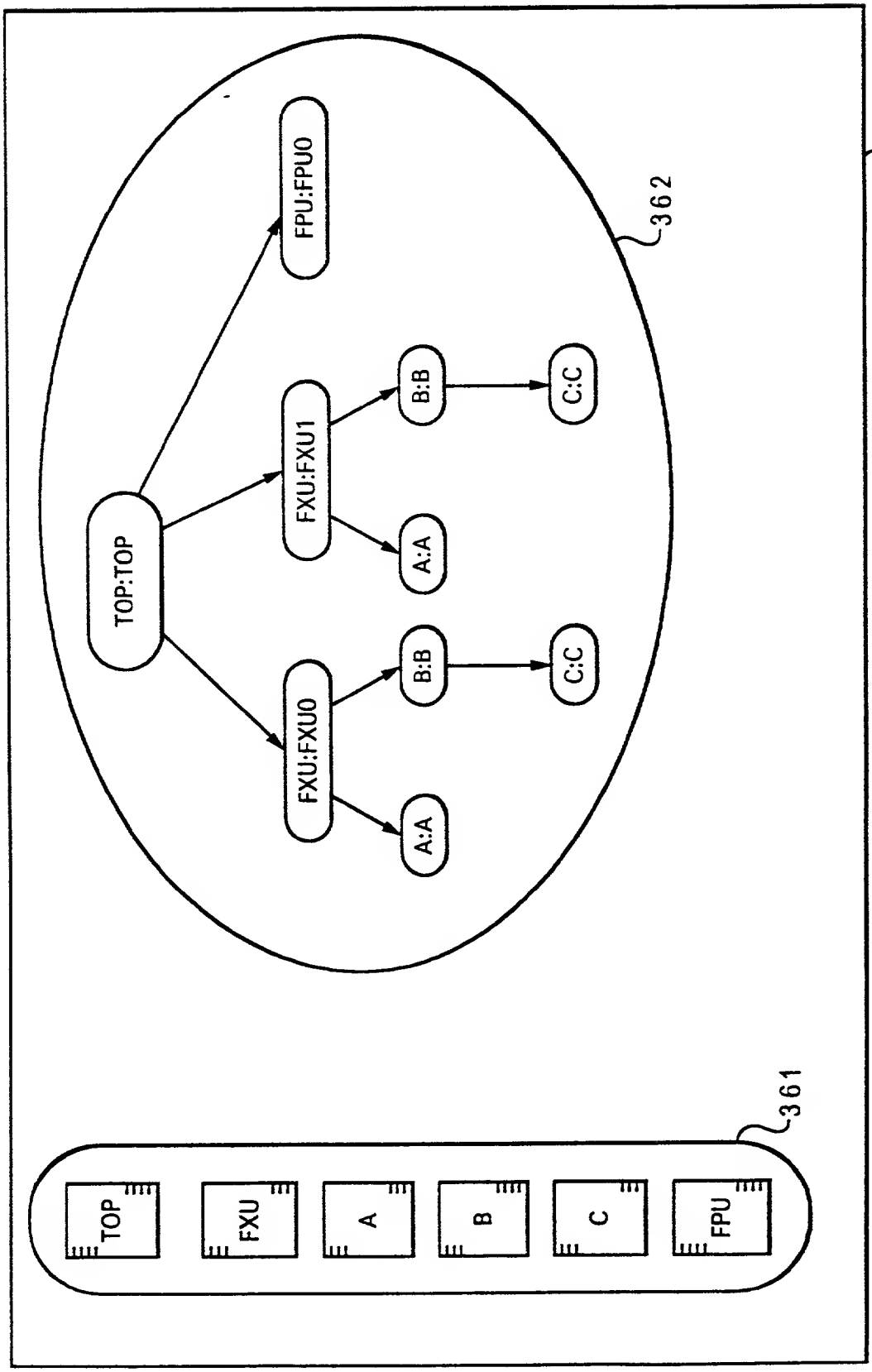

TOP
FXU
A
B
C
FPU
361
TOP:TOP
FPU:FPU0
FXU:FXU1
FXU:FXU0
A:A
B:B
C:C
A:A
B:B
C:C
362
44

Fig. 3D

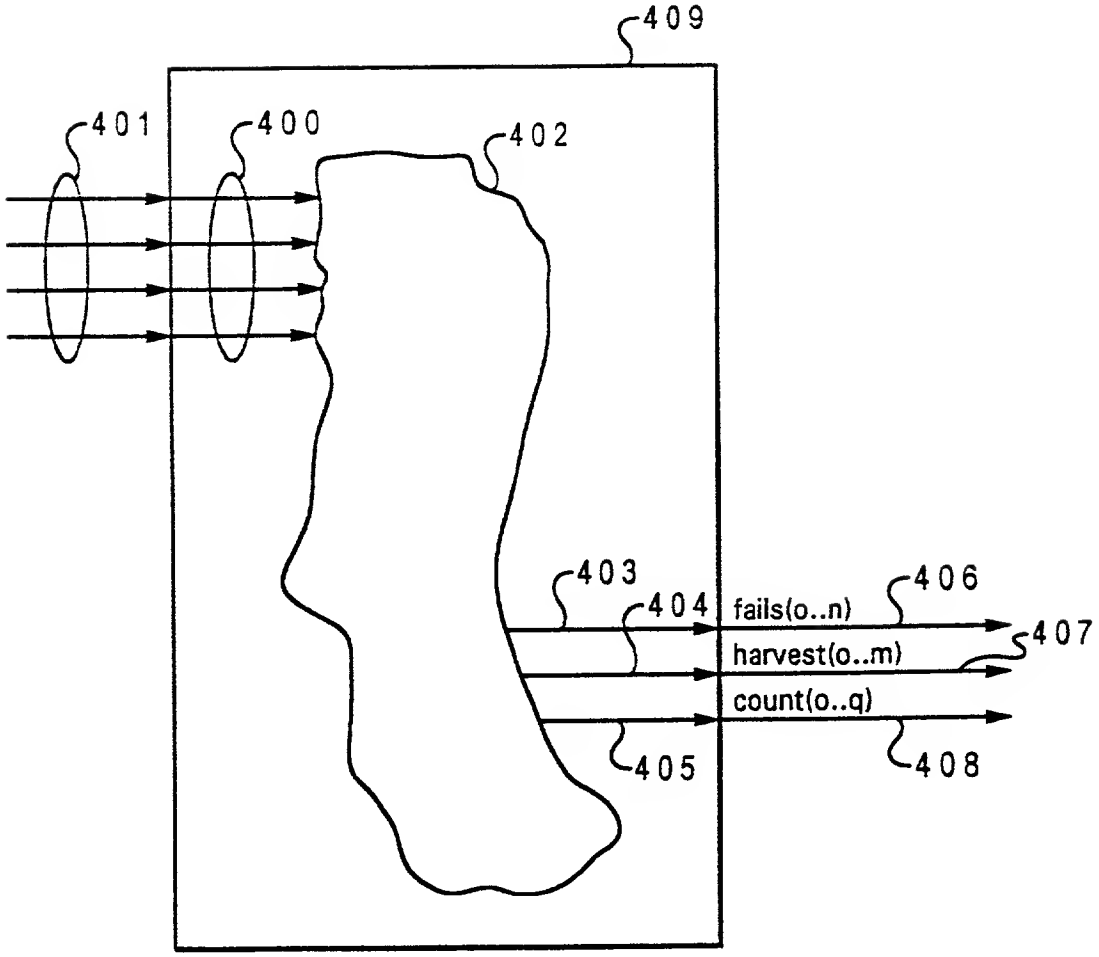

409
401
400
402
403
404
fails(o..n)
406
407
harvest(o..m)
count(o..q)
405
408

Fig. 4A

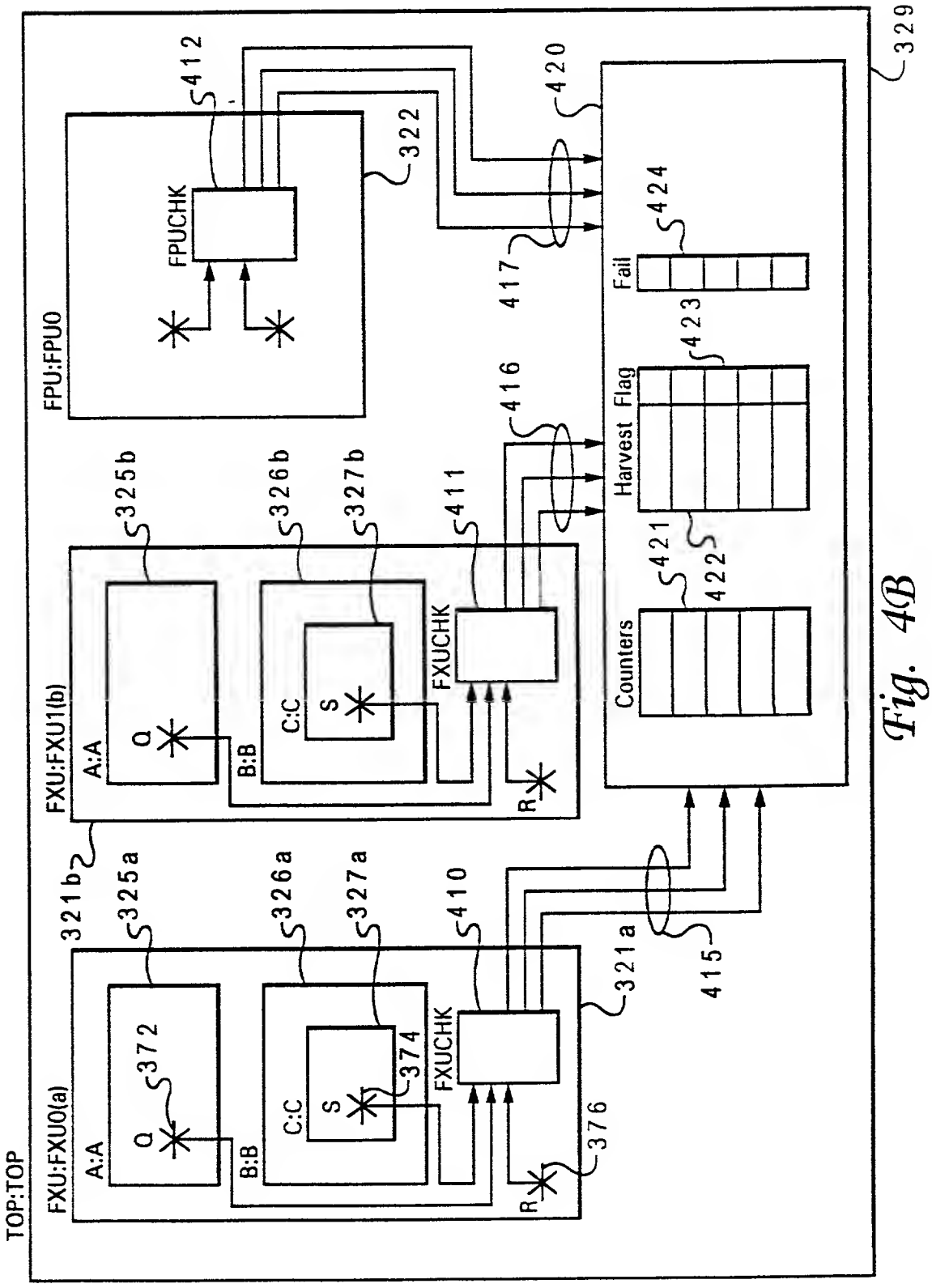

TOP:TOP
FXU:FXU0(a)
A:A
Q
372
B:B
C:C
S
374
FXUCHK
376
R
325a
326a
327a
410
321b
321a
415
FXU:FXU1(b)
325b
326b
327b
411
416
FPU:FPU0
FPUCHK
412
322
417
420
329
Harvest Flag
423
Fail
424
Counters
421
422

Fig. 4B

```
ENTITY FXUCHK IS

    PORT(   S_IN        :     IN std_ulogic;
          - Q_IN        :     IN std_ulogic;
            R_IN        :     IN std_ulogic;
            clock       :     IN std_ulogic;
            fails       :     OUT std_ulogic_vector(0 to 1);
            counts      :     OUT std_ulogic_vector(0 to 2);
            harvests    :     OUT std_ulogic_vector(0 to 1);
        );

452 --!! BEGIN
    --!! Design Entity: FXU;

453 --!! Inputs
    --!! S_IN        = >     B.C.S;
    --!! Q_IN        = >     A.Q;
    --!! R_IN        = >     R;
    --!! CLOCK       = >     clock;
    --!! End Inputs

454 --!! Fail Outputs;
    --!! 0 : "Fail message for failure event 0";
    --!! 1 : "Fail message for failure event 1";
    --!! End Fail Outputs;

455 --!! Count Outputs;
    --!! 0 : <event0> clock;
    --!! 1 : <event1> clock;
    --!! 2 : <event2> clock;
    --!! End Count Outputs;

456 --!! Harvest Outputs;
    --!! 0 : "Message for harvest event 0";
    --!! 1 : "Message for harvest event 1";
    --!! End Harvest Outputs;

457 --!! End;

ARCHITECTURE example of FXUCHK IS

BEGIN

    ... HDL code for entity body section ...

END;
```

450

451

458

440

Fig. 4C

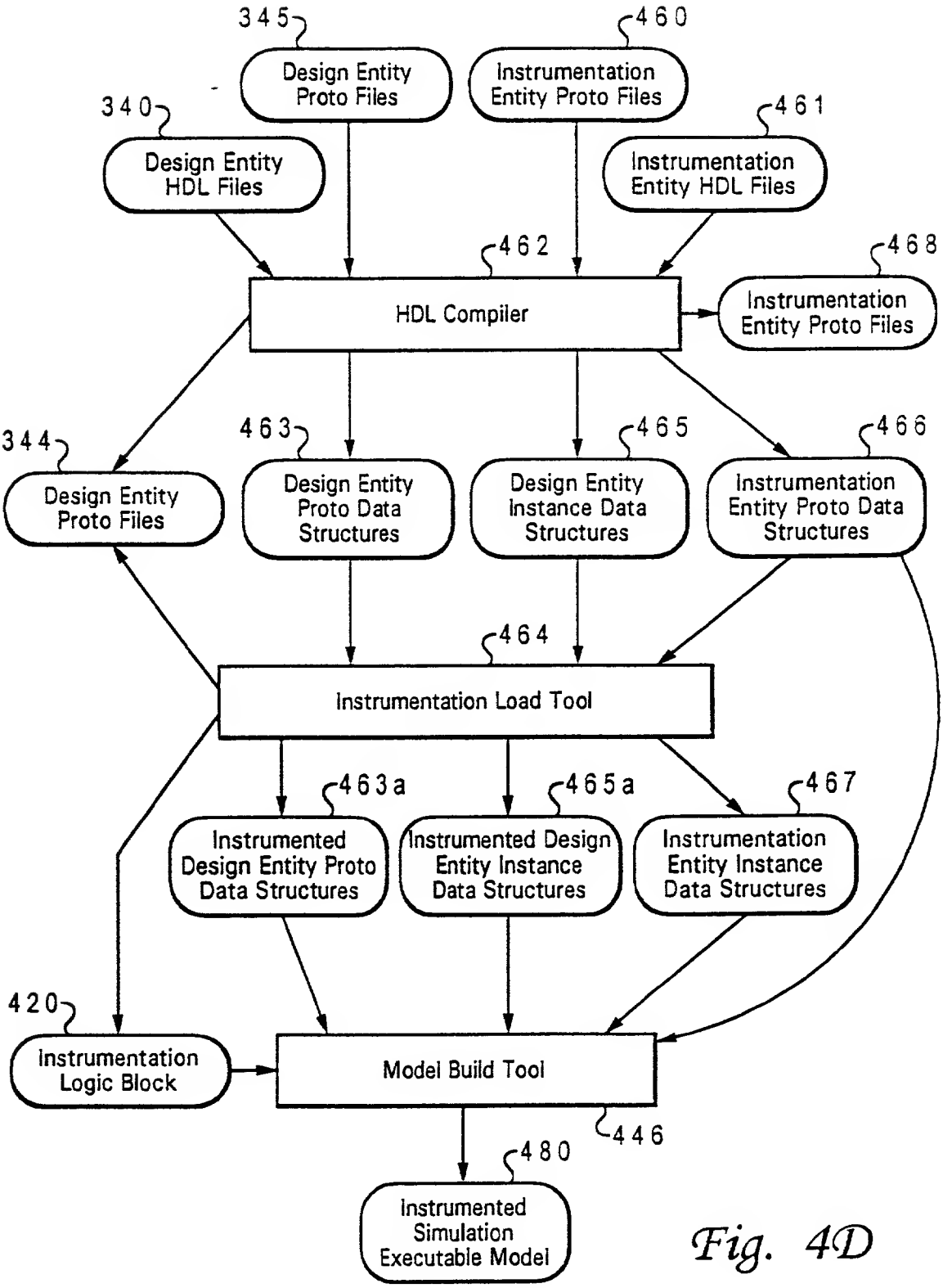

345
Design Entity Proto Files
460
Instrumentation Entity Proto Files
340
Design Entity HDL Files
461
Instrumentation Entity HDL Files
462
HDL Compiler
468
Instrumentation Entity Proto Files
344
Design Entity Proto Files
463
Design Entity Proto Data Structures
465
Design Entity Instance Data Structures
466
Instrumentation Entity Proto Data Structures
464
Instrumentation Load Tool
463a
Instrumented Design Entity Proto Data Structures
465a
Instrumented Design Entity Instance Data Structures
467
Instrumentation Entity Instance Data Structures
420
Instrumentation Logic Block
Model Build Tool
446
480
Instrumented Simulation Executable Model

Fig. 4D

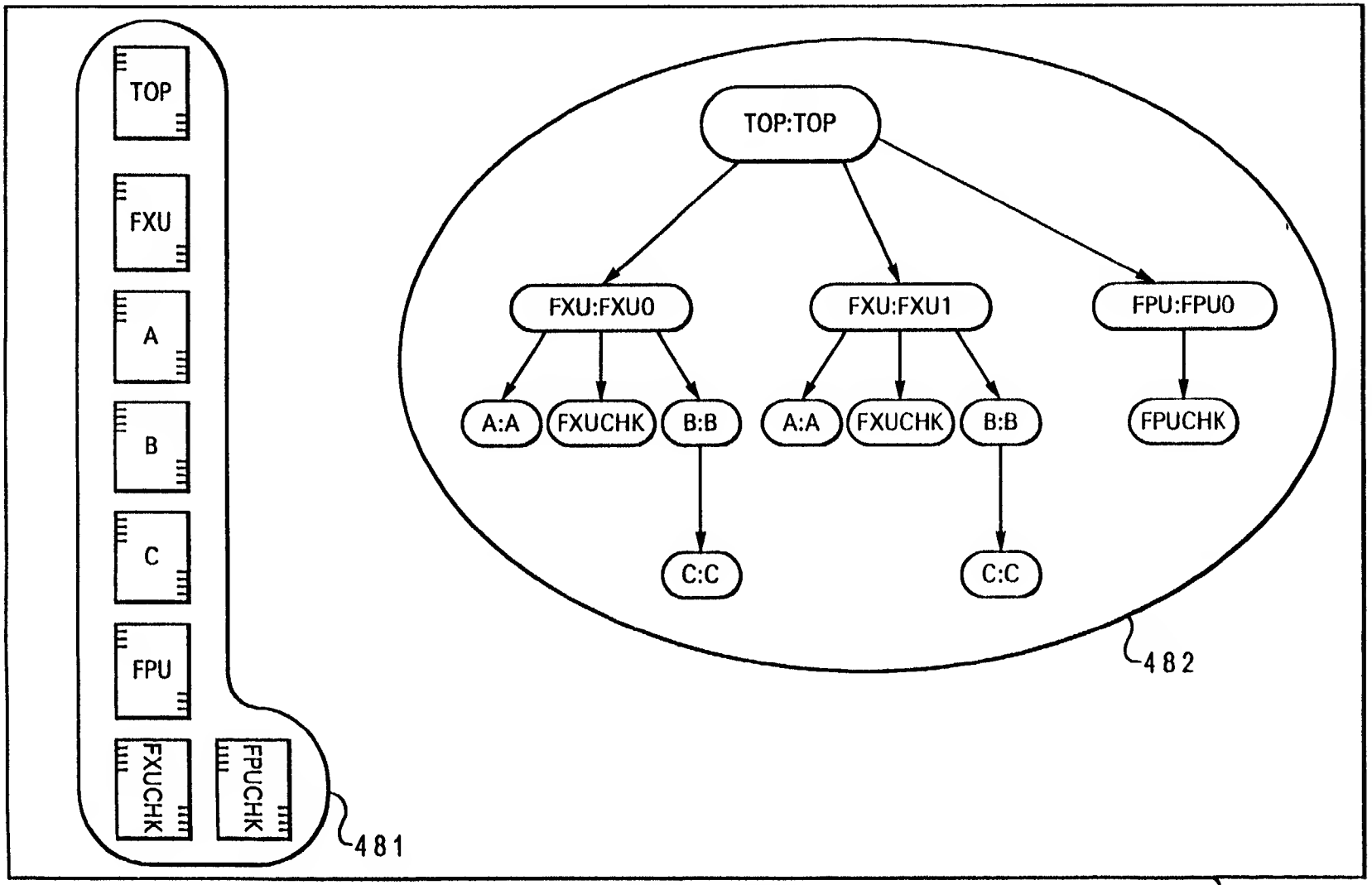

TOP
FXU
A
B
C
FPU
FXUCHK
FPUCHK
481
TOP:TOP
FXU:FXU0
FXU:FXU1
FPU:FPU0
A:A
FXUCHK
B:B
A:A
FXUCHK
B:B
FPUCHK
C:C
C:C
482
44

Fig. 4E

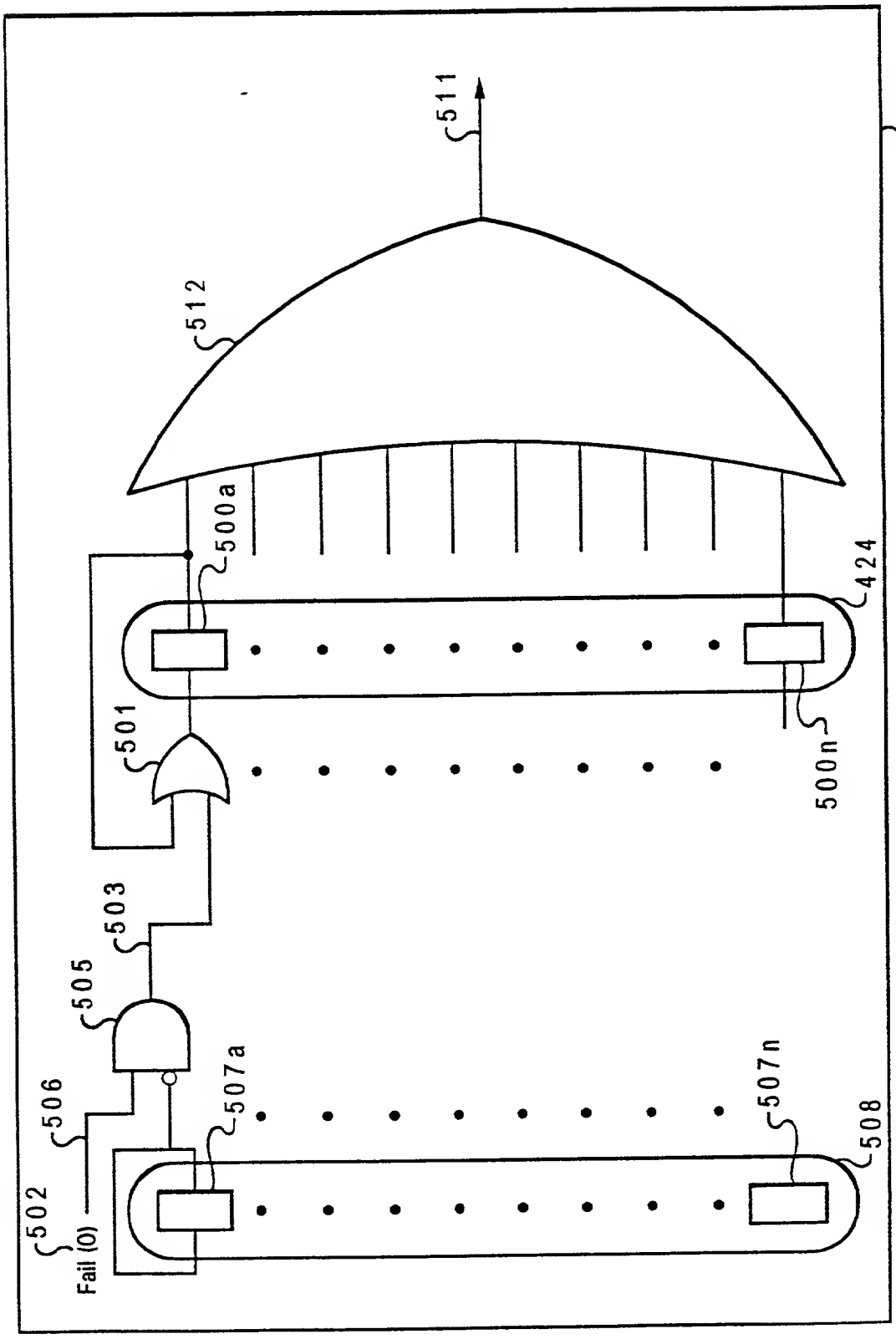

511
512
420
500a
424
501
500n
503
505
506
507a
507n
508
502
Fail (0)

Fig. 5A

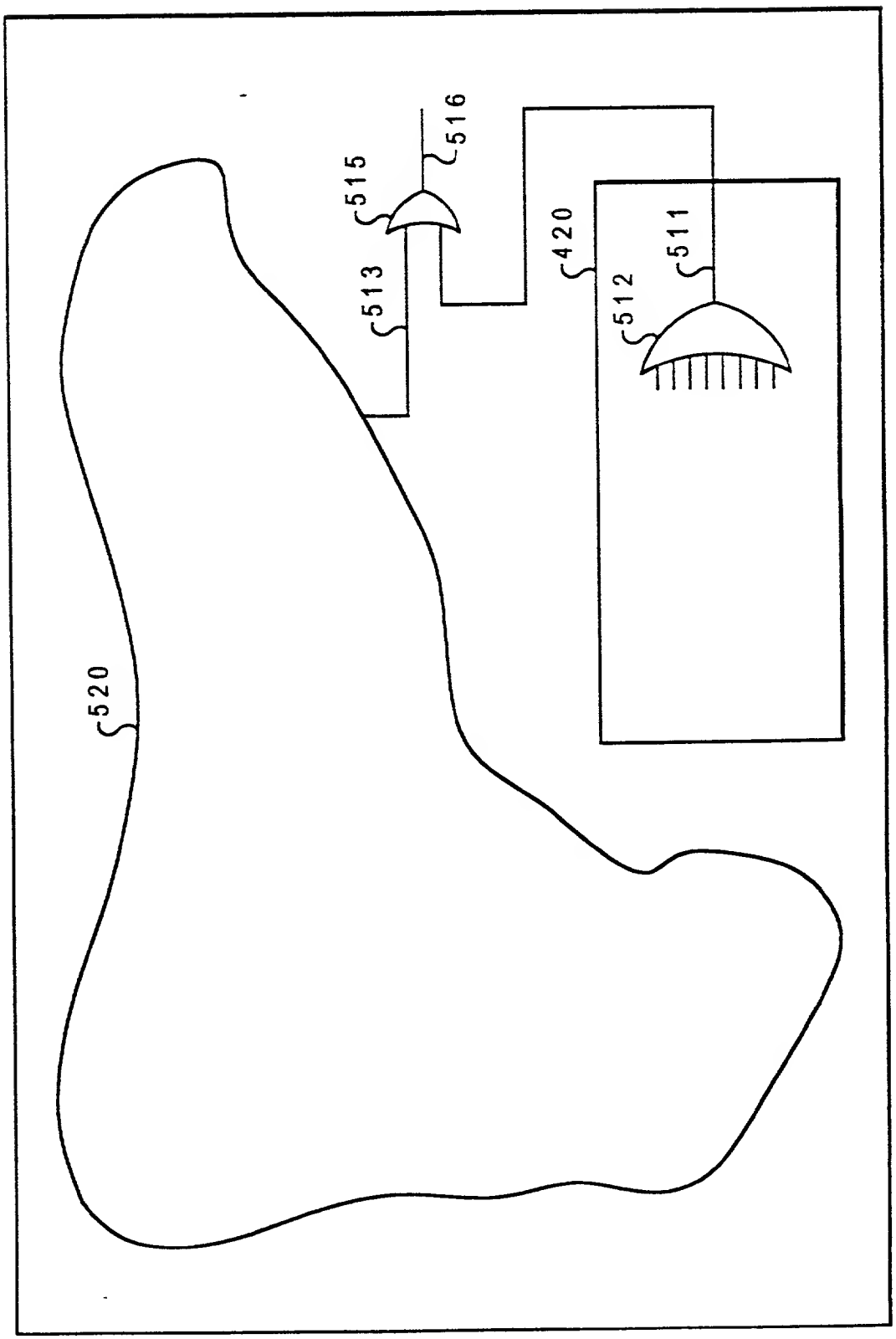


Fig. 5B

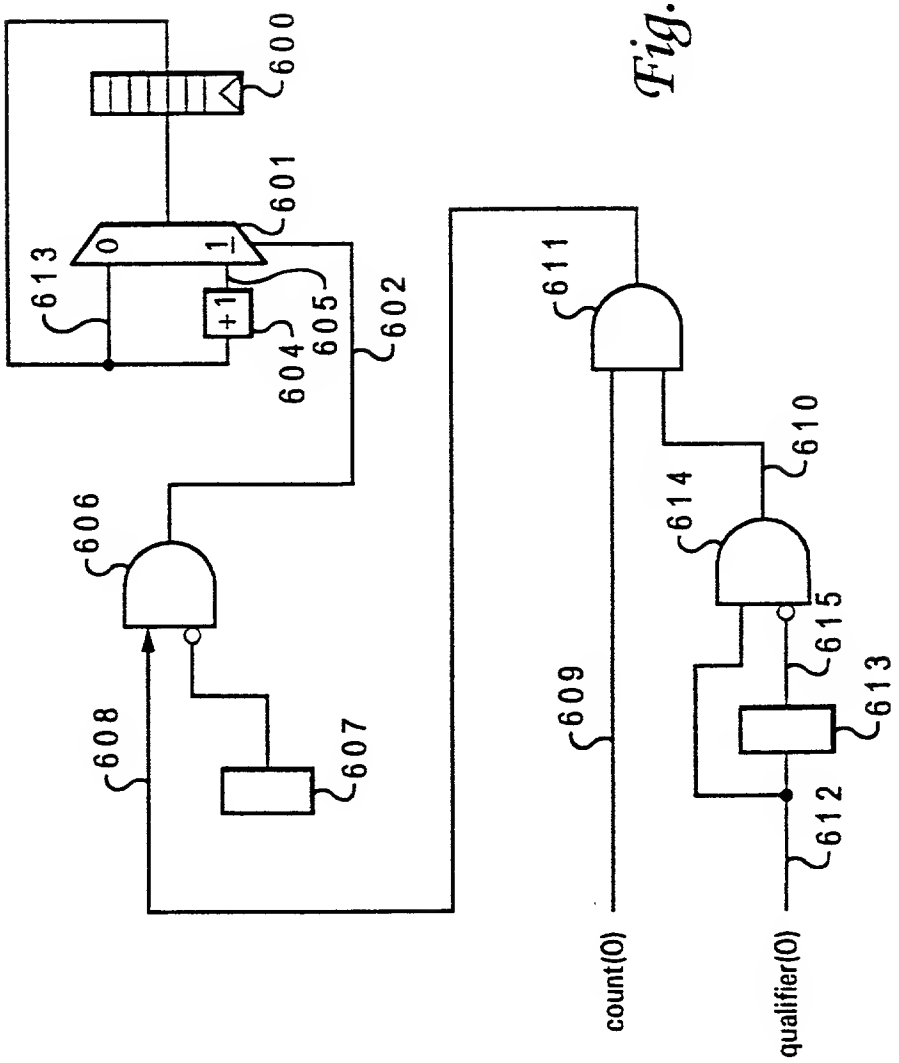

600
601
0
1
613
+1
604
605
602
606
608
607
611
609
614
610
615
613
612
count(0)
qualifier(0)

Fig. 6A

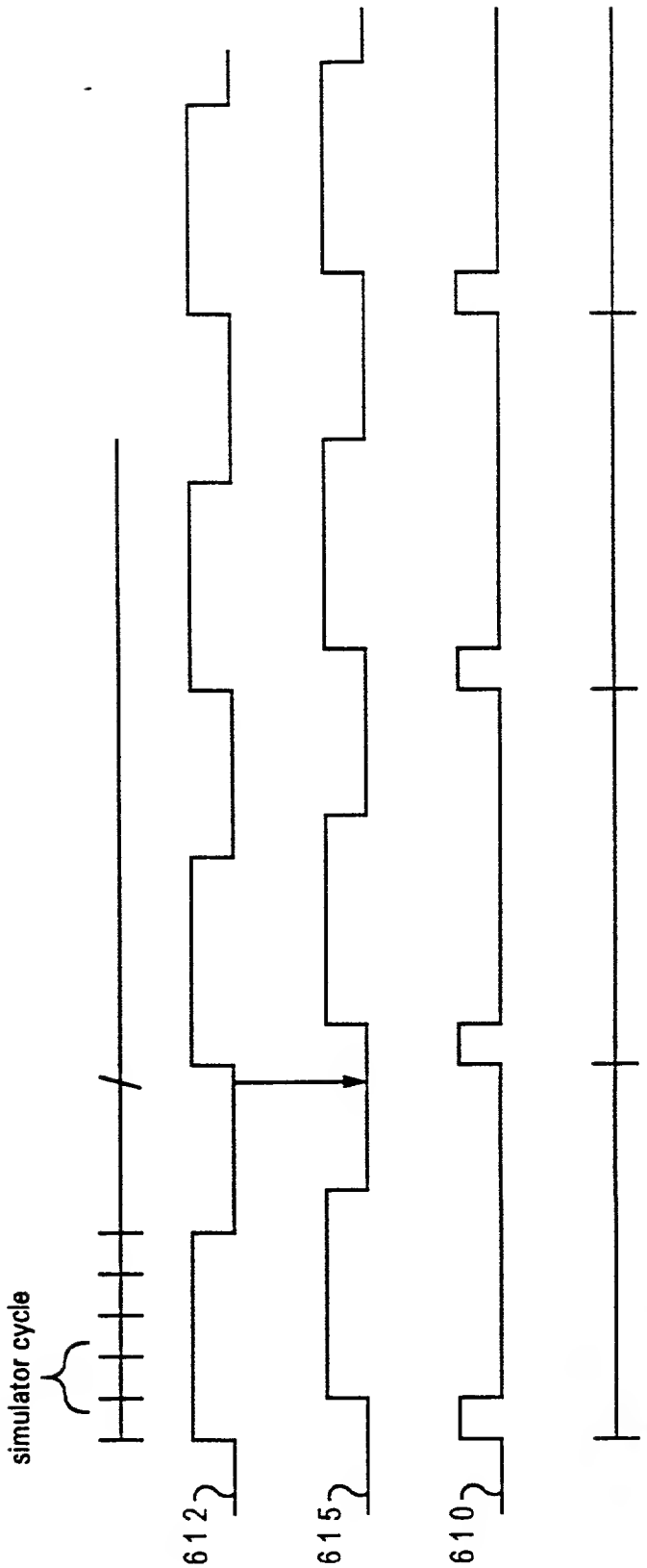


Fig. 6B

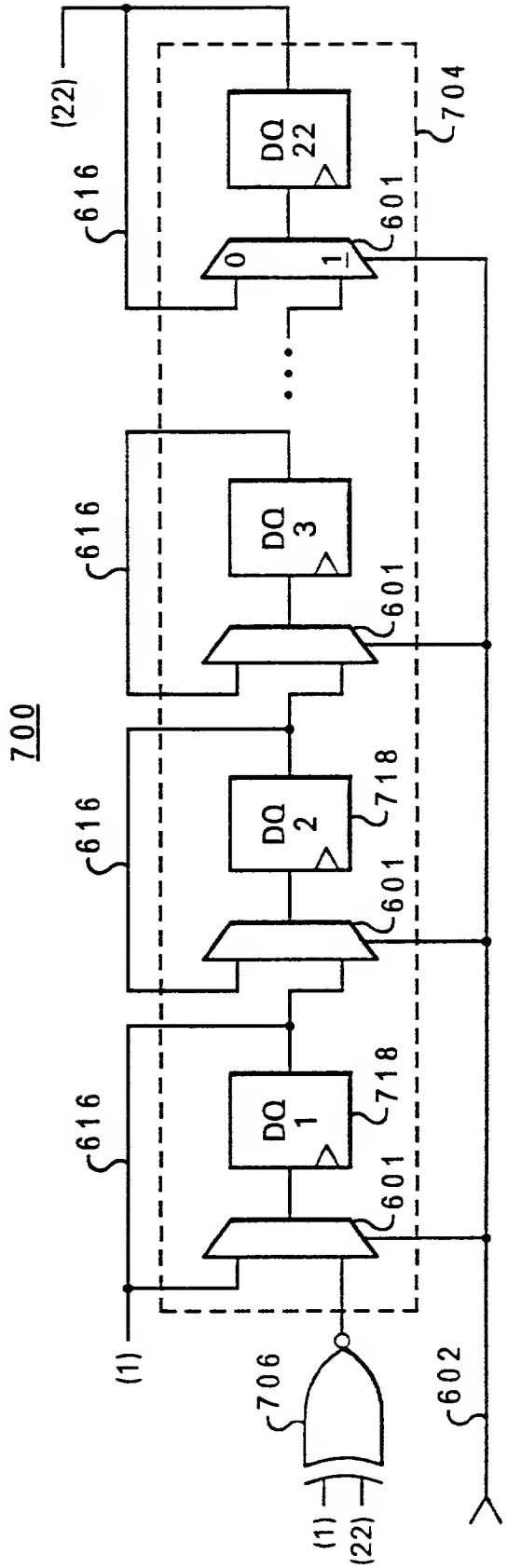


Fig. 7

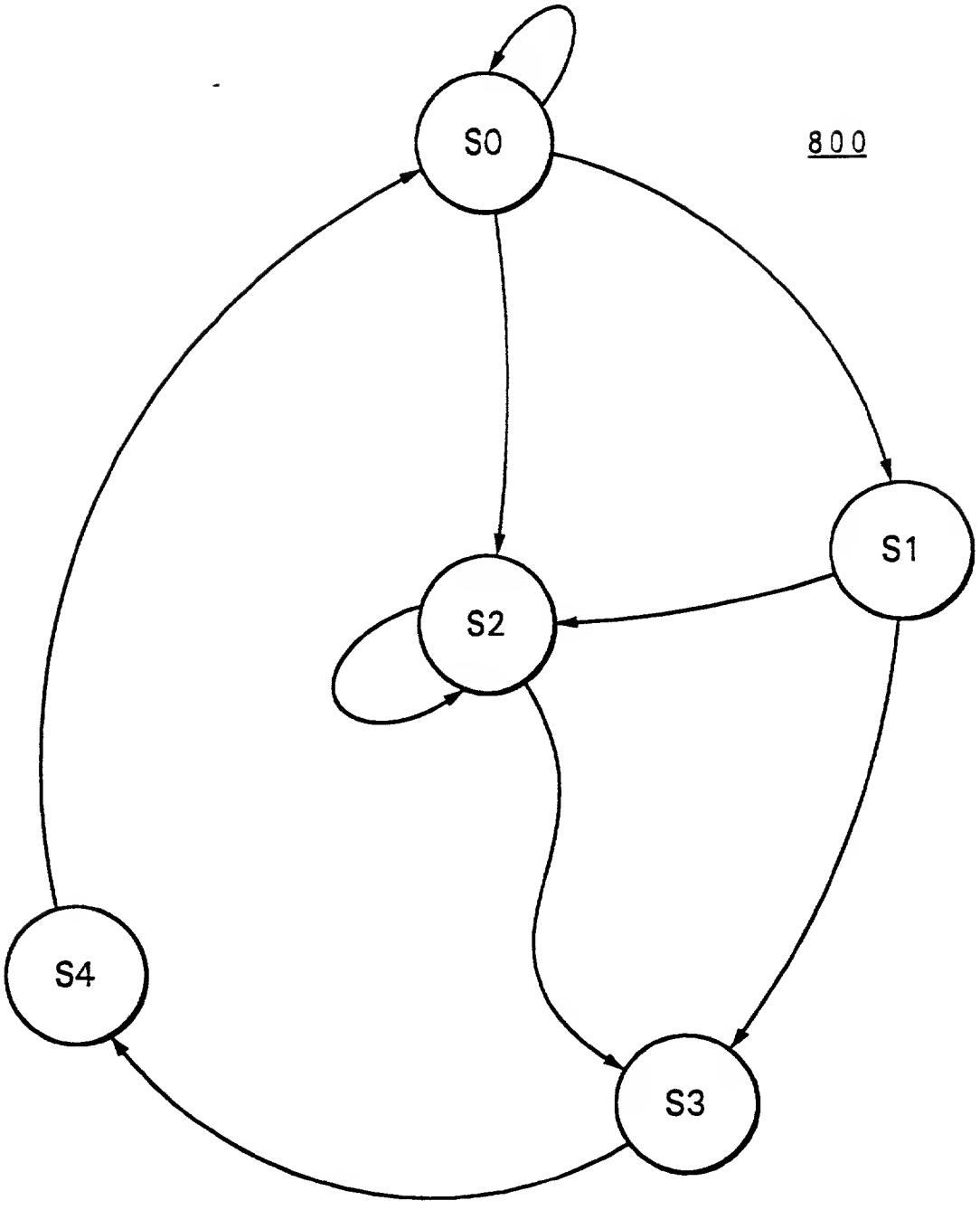


Fig. 8A
Prior Art

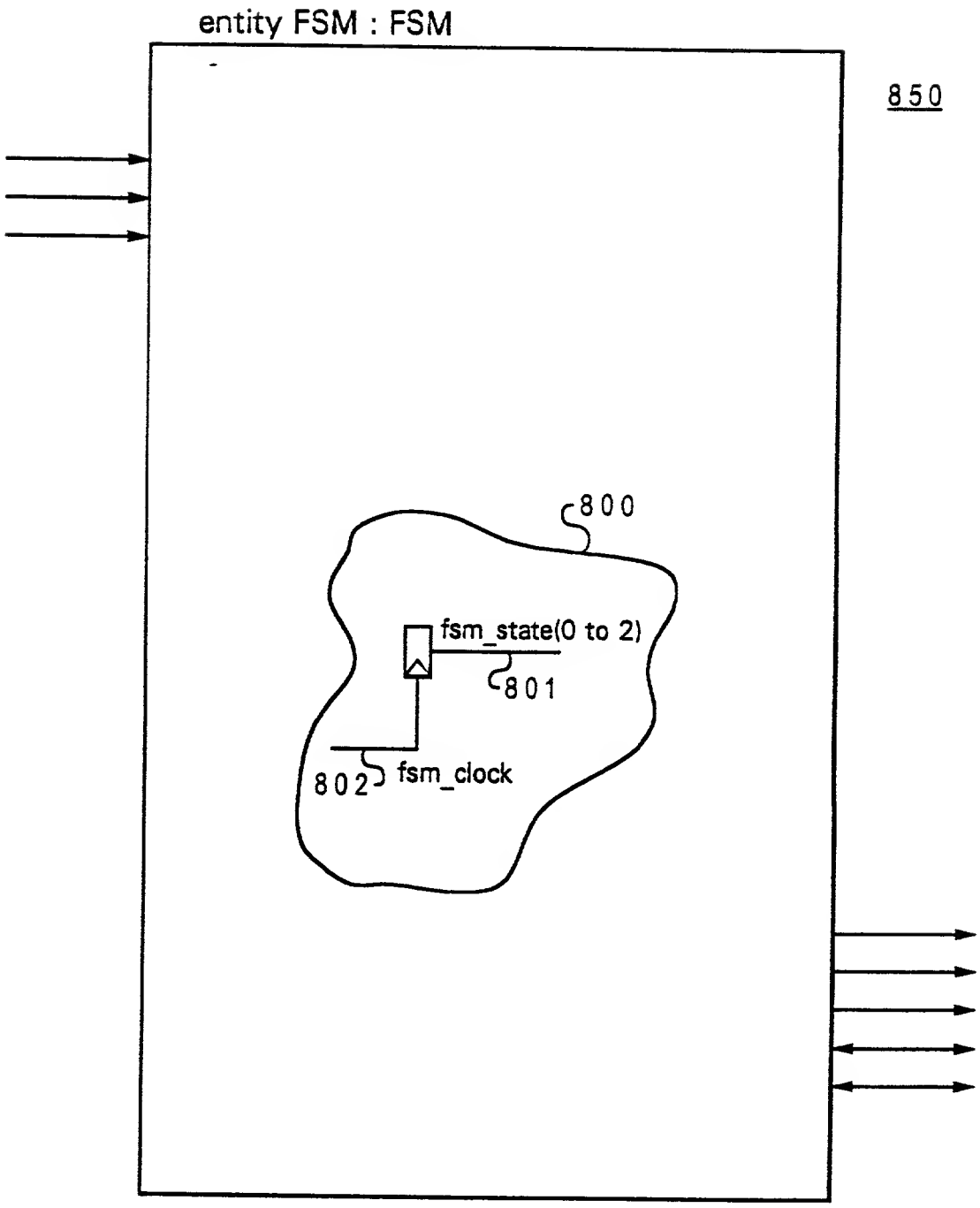


Fig. 8B
Prior Art

```
ENTITY FSM IS

    PORT(
            ....ports for entity fsm....
        );

    ARCHITECTURE FSM OF FSM IS

    BEGIN

            ... HDL code for FSM and rest of the entity ...

            fsm_state(0 to 2) < = ... Signal 801 ...

853 {  --!! Embedded FSM : examplefsm;
859 {  --!! clock         : (fsm_clock);
854 {  --!! state_vector  : (fsm_state(0 to 2));
855 {  --!! states        : (S0, S1, S2, S3, S4);
856 {  --!! state_encoding : ('000', '001', '010', '011', '100');
       --!! arcs          : (S0 => S0, S0 => S1, S0 => S2,
857 {  --!!                  (S1 => S2, S1 => S3, S2 => S2,
       --!!                  (S2 => S3, S3 => S4, S4 => S0);
858 {  --!! End FSM;

                ,
                ,
                ,
                ,
                ,
                ,
                ,

    END;
```

852 860

Fig. 8C

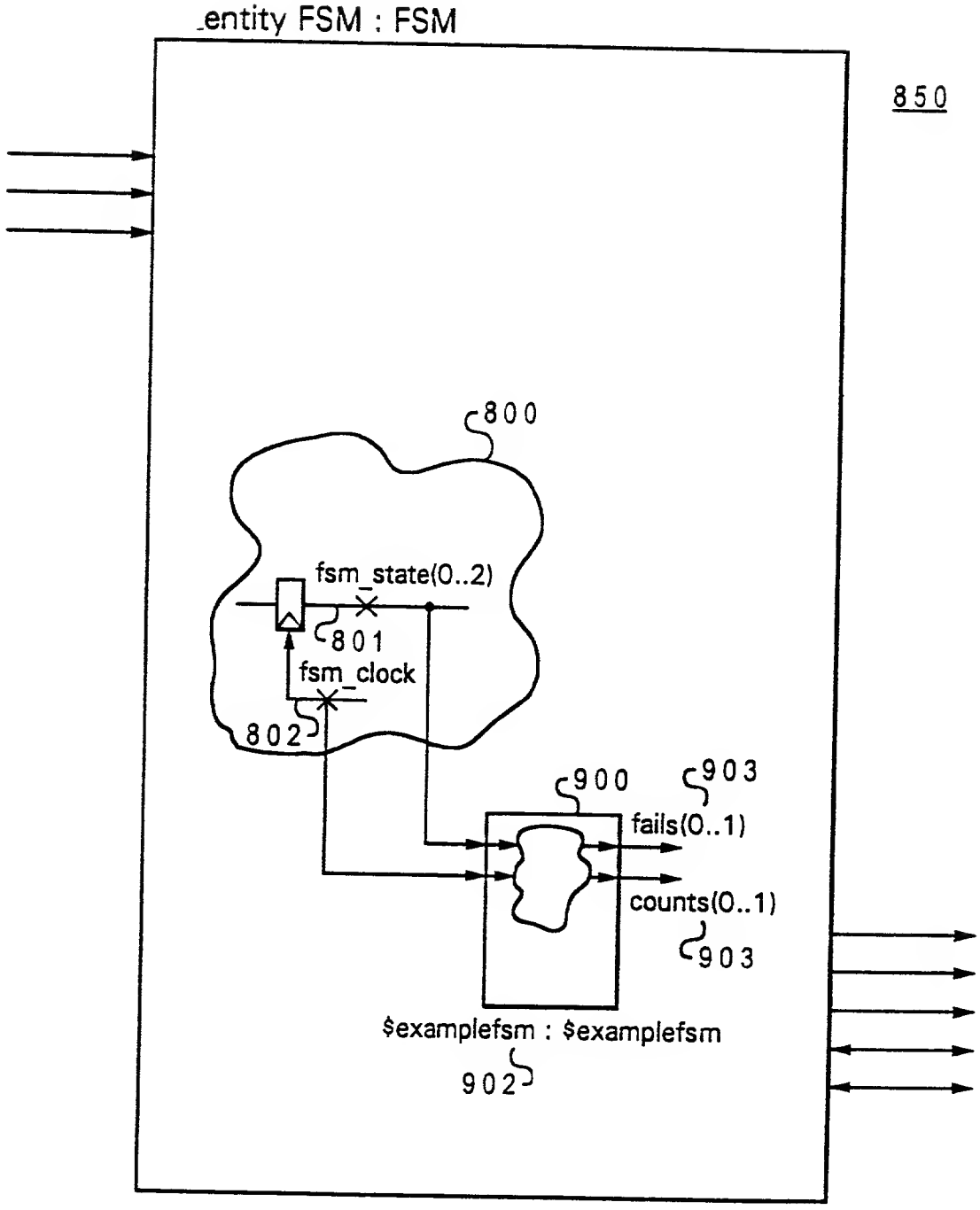
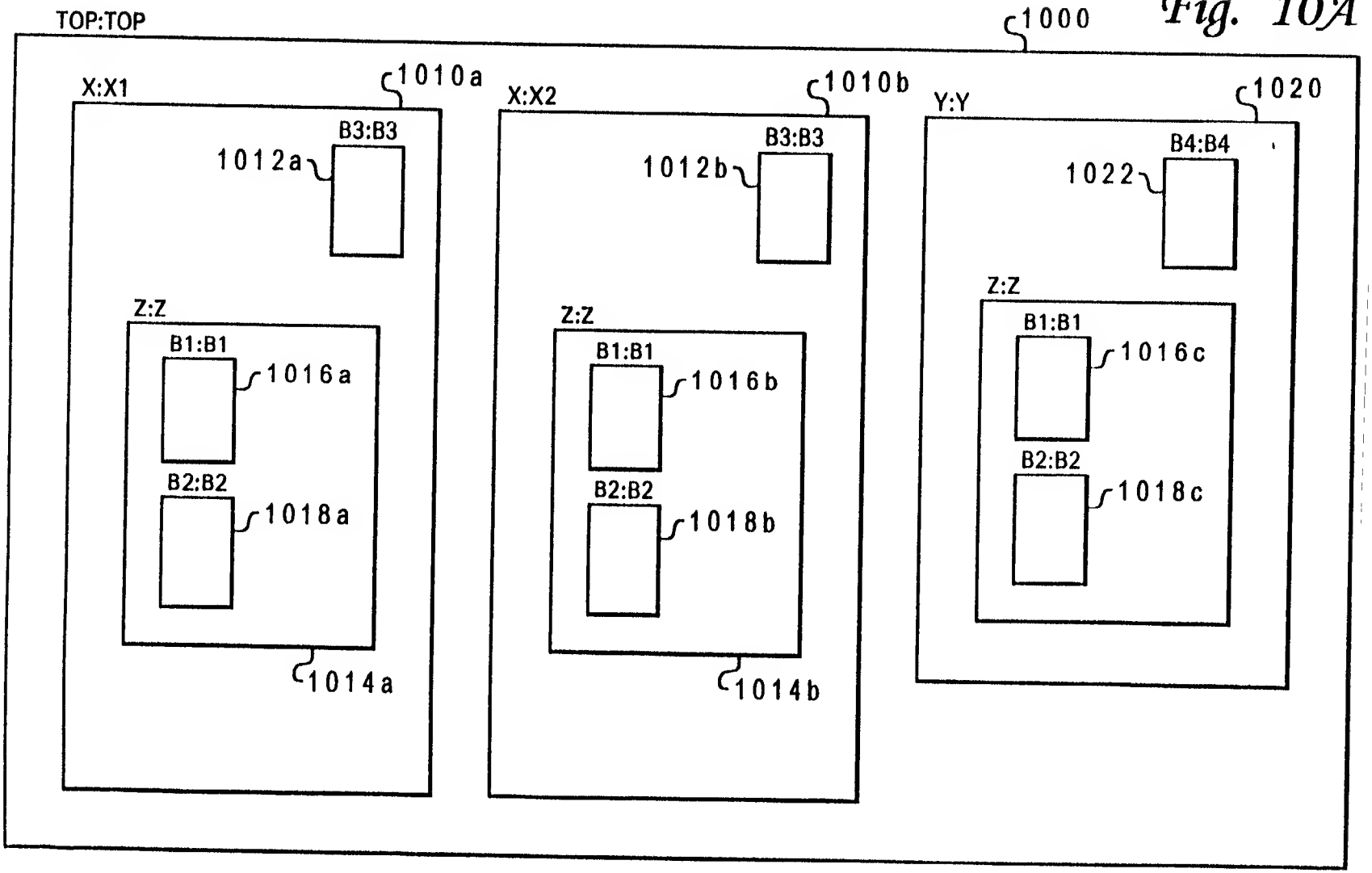

entity FSM : FSM
850
800
fsm_state(0..2)
801
fsm_clock
802
900
903
fails(0..1)
counts(0..1)
903
$examplefsm : $examplefsm
902

Fig. 9

Fig. 10A


TOP:TOP
1000
X:X1
1010a
B3:B3
1012a
Z:Z
B1:B1
1016a
B2:B2
1018a
1014a
X:X2
1010b
B3:B3
1012b
Z:Z
B1:B1
1016b
B2:B2
1018b
1014b
Y:Y
1020
B4:B4
1022
Z:Z
B1:B1
1016c
B2:B2
1018c

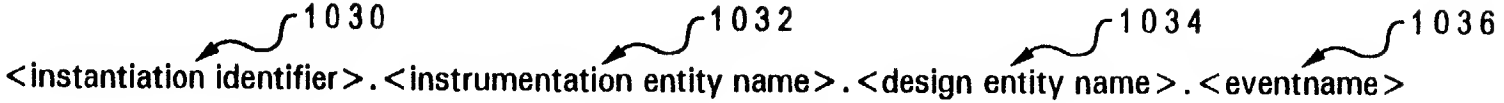


Fig. 10B

1030	1032	1034	1036	
X1	B3	X	COUNT1	1040
X1.Z	B1	Z	COUNT1	1041
X1.Z	B2	Z	COUNT1	1042
X2	B3	X	COUNT1	1043
X2.Z	B1	Z	COUNT1	1044
X2.Z	B2	Z	COUNT1	1045
Y	B4	Y	COUNT1	1046
Y.Z	B1	Z	COUNT1	1047
Y.Z	B2	Z	COUNT1	1048

Fig. 10C

Fig. 10D

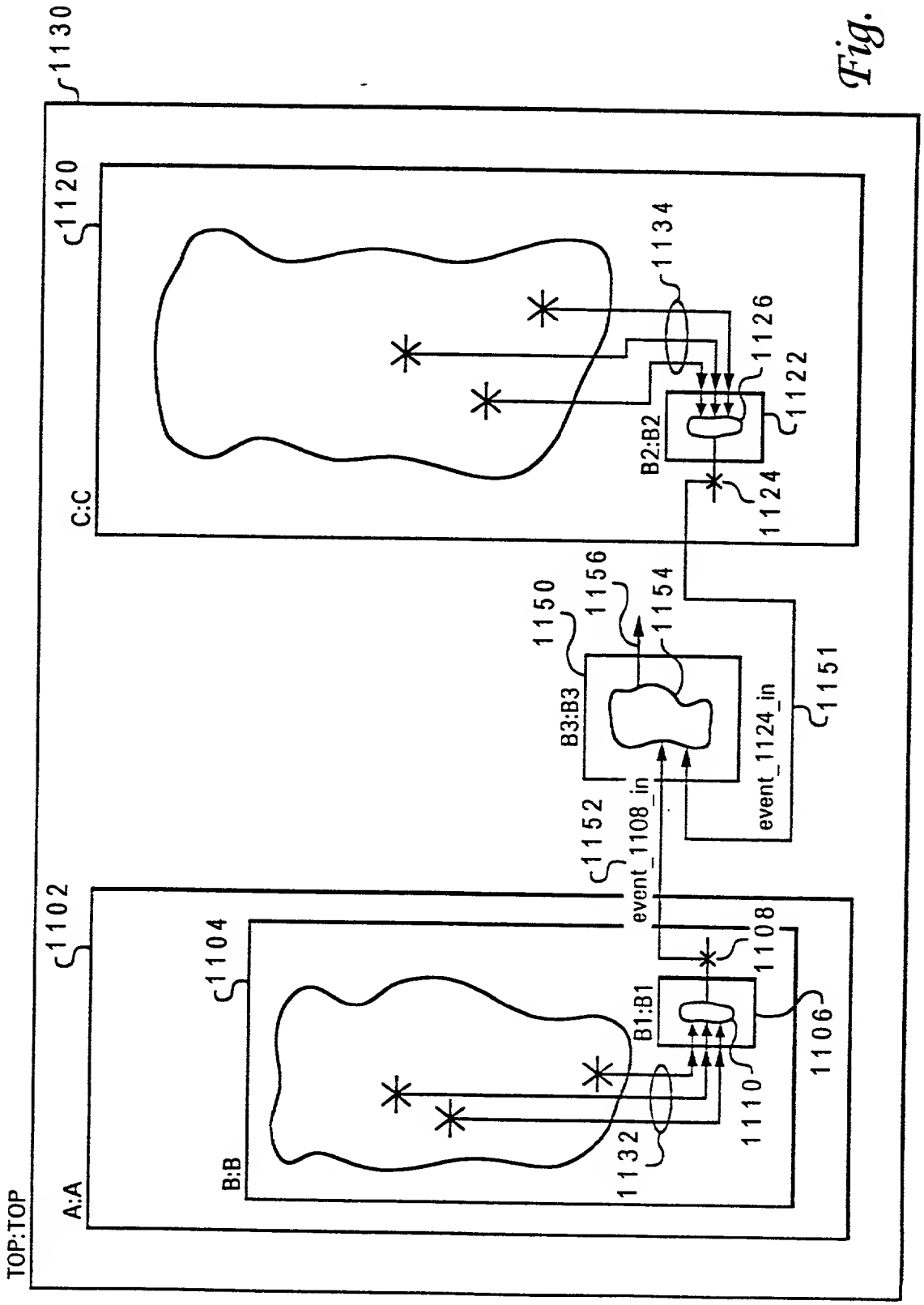

TOP:TOP
1130
A:A
1102
B:B
1104
C:C
1120
1132
1110
1106
B1:B1
1108
event_1108_in
1152
1150
B3:B3
1156
1154
1151
event_1124_in
1134
1126
B2:B2
1122
1124

Fig. 11A

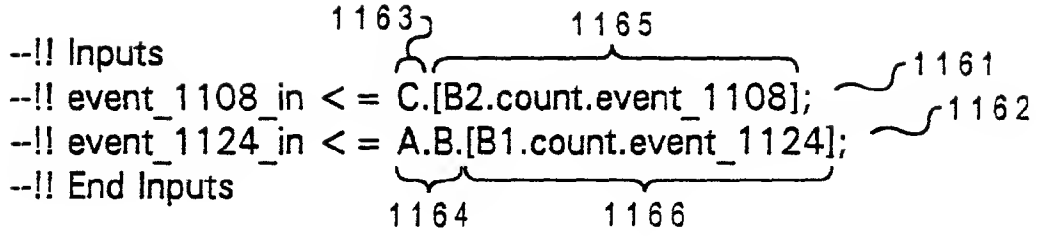


Fig. 11B

```
--!! Inputs
--!! event_1108_in < = C.[count.event_1108];
--!! event_1124_in < = B.[count.event_1124];
--!! End Inputs
```

1171
1172

Fig. 11C

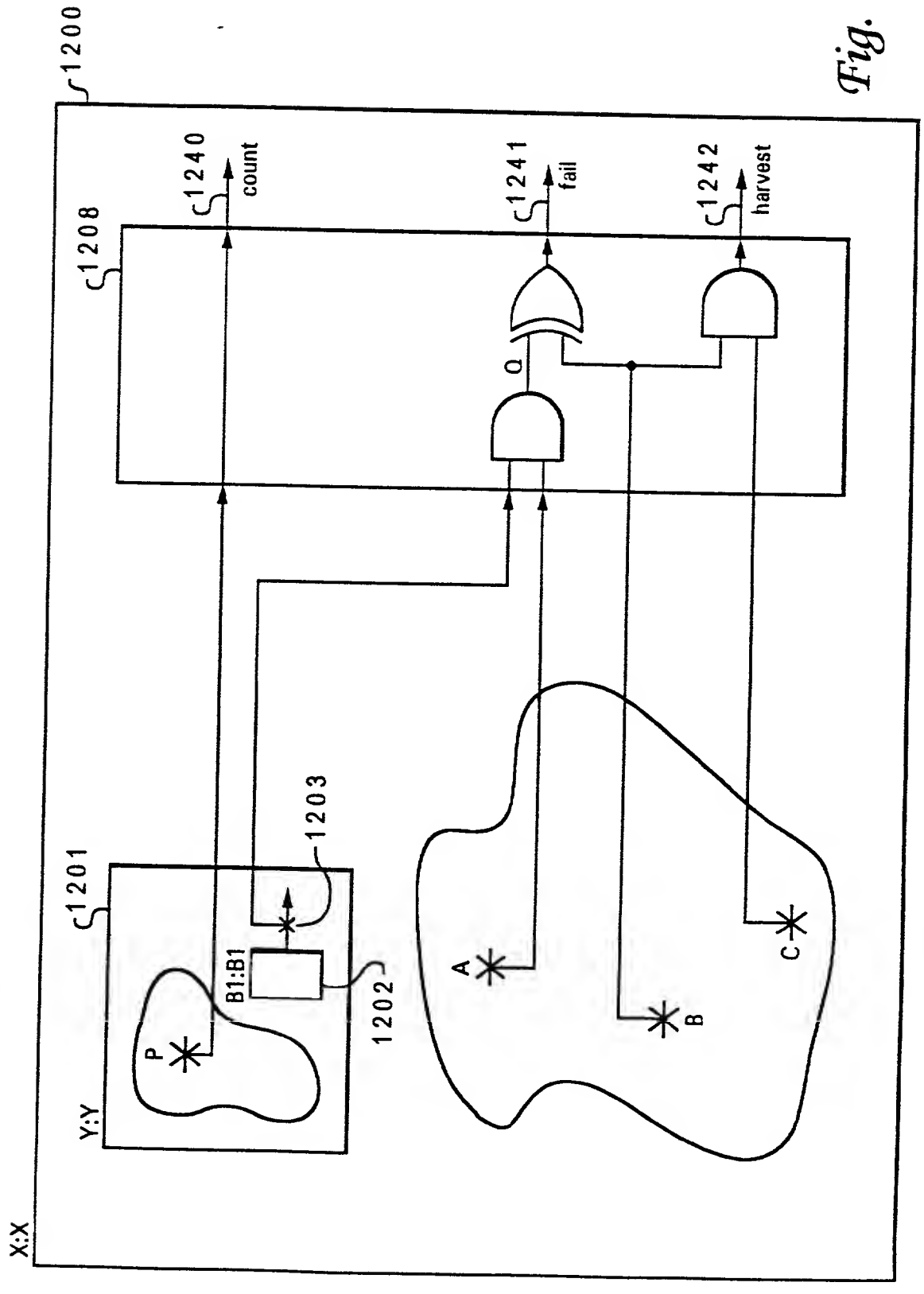

1200
1208
1240
count
1241
fail
1242
harvest
Q
1201
1203
B1:B1
1202
P
Y:Y
A
B
C
X:X

Fig. 12A

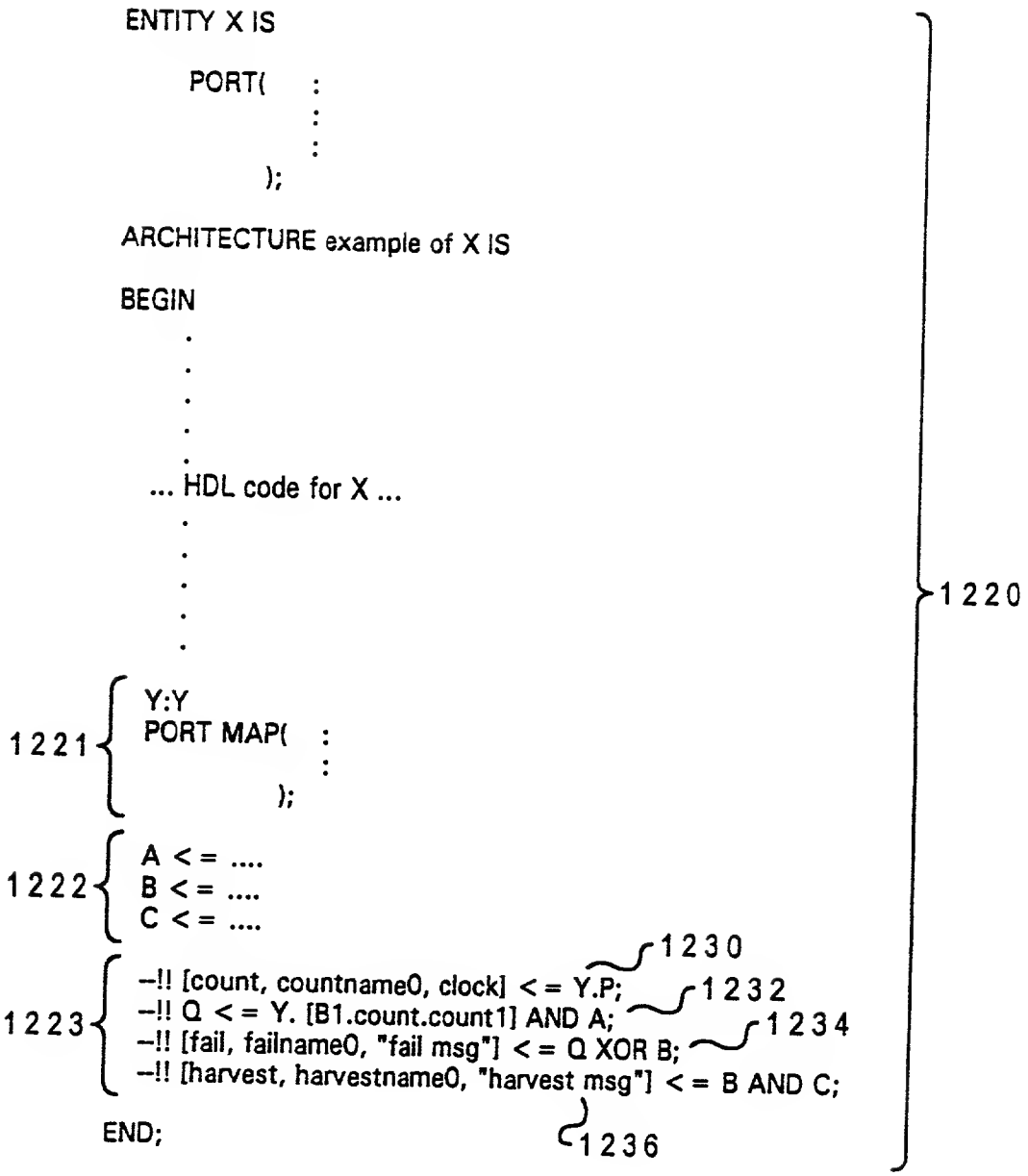

ENTITY X IS
PORT(:
:
:
);
ARCHITECTURE example of X IS
BEGIN
.
.
.
.
.
... HDL code for X ...
.
.
.
.
.
1220
1221
Y:Y
PORT MAP(:
:
);
1222
A < = ...
B < = ...
C < = ...
1223
1230
1232
1234
--!! [count, countname0, clock] < = Y.P;
--!! Q < = Y. [B1.count.count1] AND A;
--!! [fail, failname0, "fail msg"] < = Q XOR B;
--!! [harvest, harvestname0, "harvest msg"] < = B AND C;
END;
1236

Fig. 12B

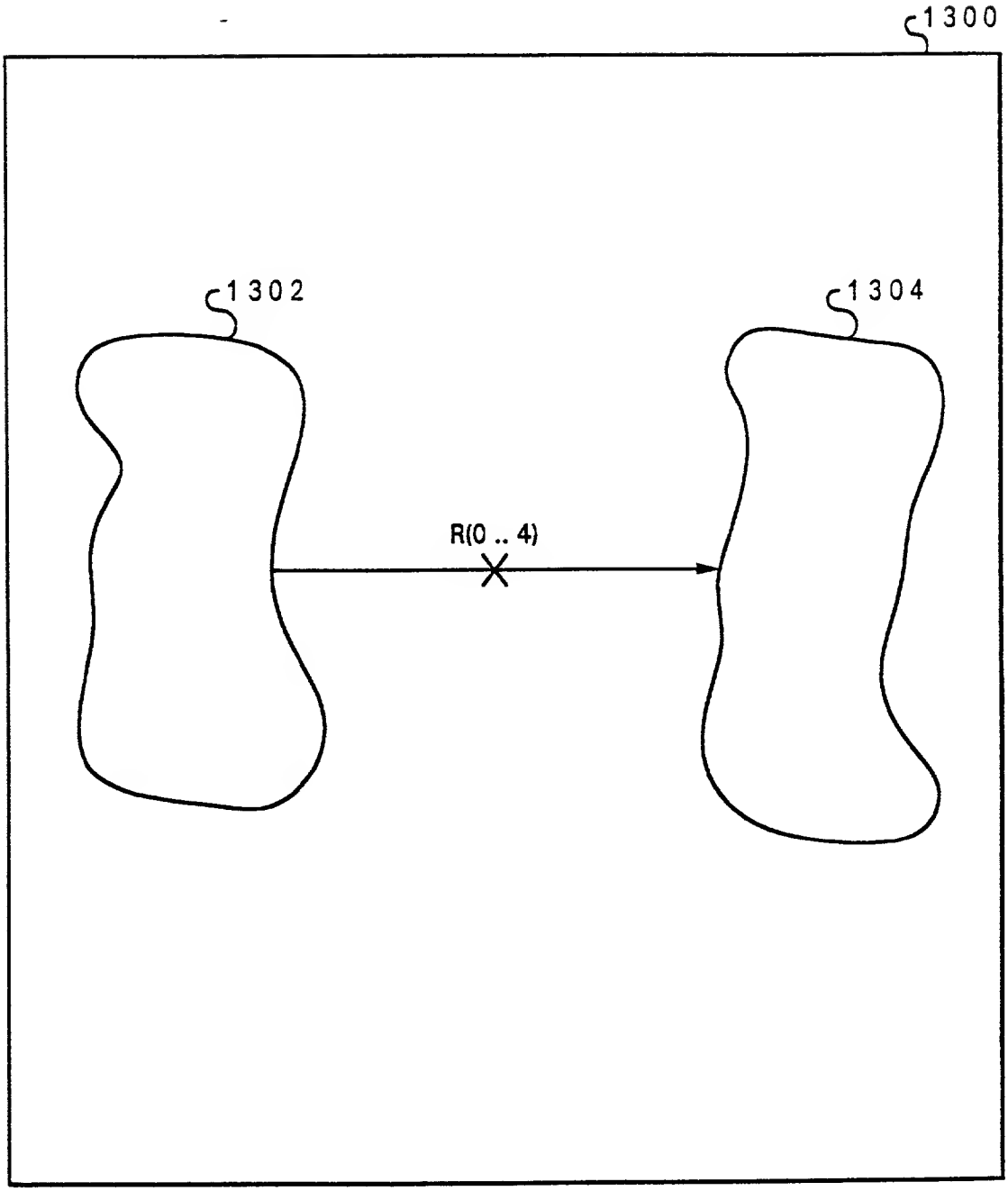


Fig. 13A

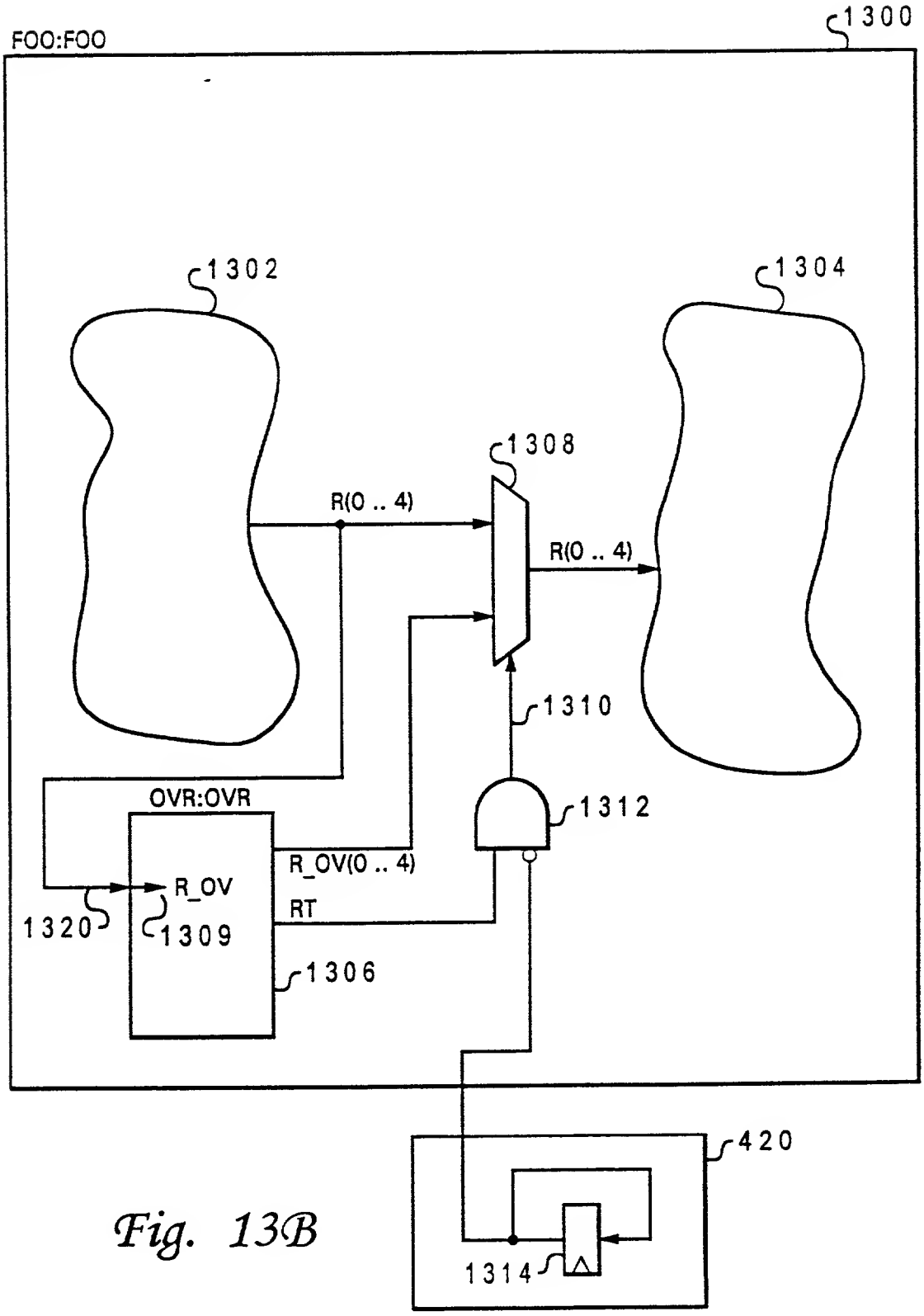

1300
FOO:FOO
1302
1304
1308
R(0 .. 4)
R(0 .. 4)
1310
1312
OVR:OVR
R_OV(0 .. 4)
RT
R_OV
1320
1309
1306
420
1314

Fig. 13B

```
ENTITY OVR IS                                          1364
    PORT(   R_IN        :    IN std_ulogic_vector(0 .. 4);
                .
                .
                .
            ... other ports as required ...
                .
                .                                          1362
                .
            R_OV        :    OUT std_ulogic_vector(0 .. 4);
            RT          :    OUT std_ulogic
        );                                     1363
-!! BEGIN
-!! Design Entity: FOO;

-!! Inputs (0 to 4)                          1360
-!! R_IN => {R(0 .. 4)};
-!! :
... other ports as needed ...
-!! :
-!! End Inputs
                                       1361
-!! Outputs
-!! <R_OVRRIDE> : R_OV(0 .. 4) => R(0 .. 4) [RT];
-!! End Outputs

-!! End

ARCHITECTURE example of OVR IS

BEGIN

    ... HDL code for entity body section ...

END;
```

1356 1351 1340 1358

Fig. 13C

```
ENTITY FOO IS

    PORT(    :
             :
             :
         );

ARCHITECTURE example of FOO IS

BEGIN

    .
    .
    .
    .
    .
    R <= .....
    .
    .
    .
    .
    .
```

1381

1380 {
```
    --!! R_IN <= {R};
    --!!
    --!!
    --!! R_OV(0 to 4) <= .....;
    --!! RT <= .....;
    --!! [override, R_OVRRIDE, R(0 .. 4), RT] <= R_OV(0 to 4);
```
}

1382

1383

1384

Fig. 13D

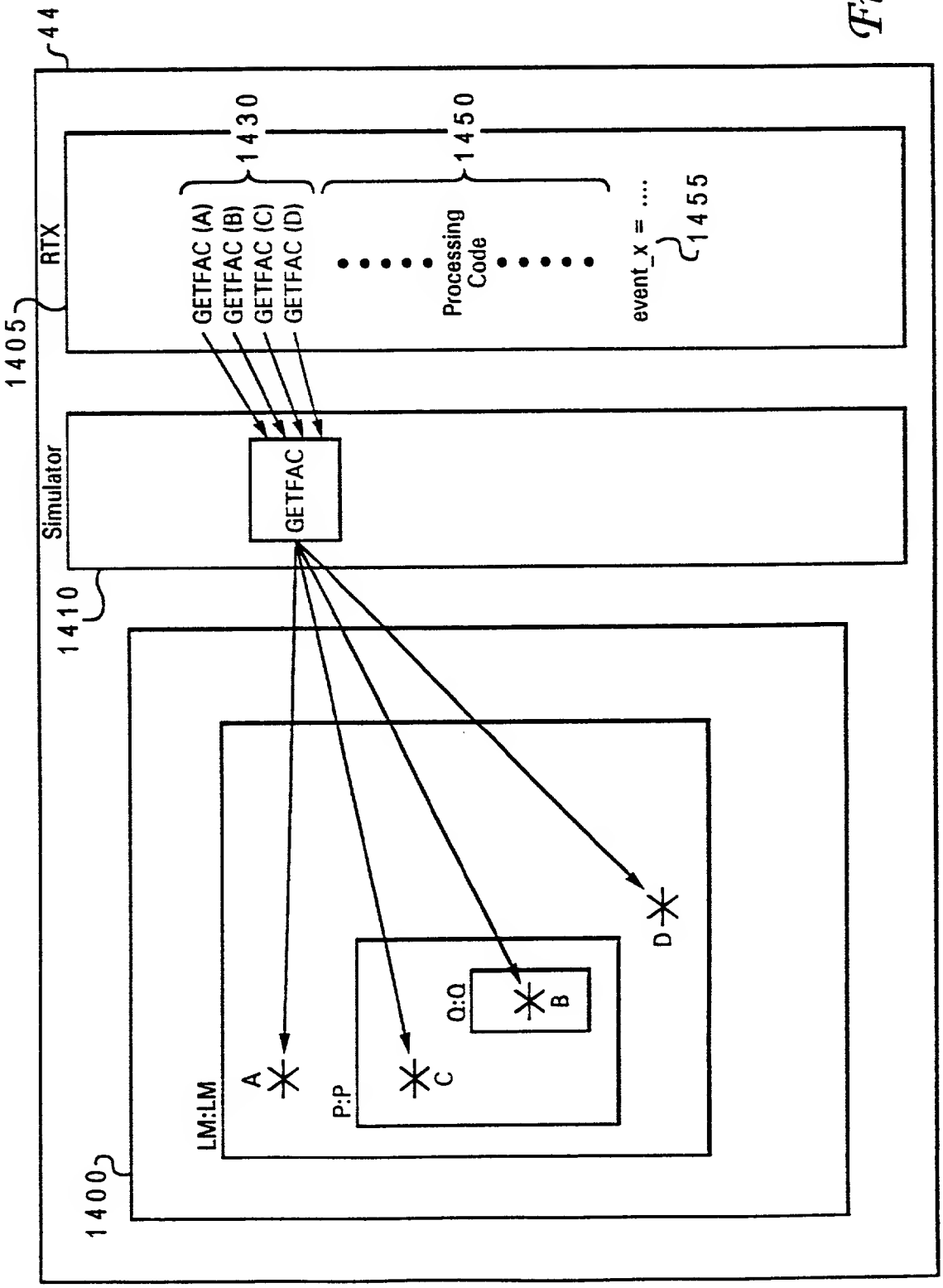

44
1405
RTX
1430
1450
GETFAC (A)
GETFAC (B)
GETFAC (C)
GETFAC (D)
Processing Code
event_x = ...
1455
Simulator
GETFAC
1410
1400
LM:LM
A
P:P
C
Q:Q
B
D

Fig. 14A

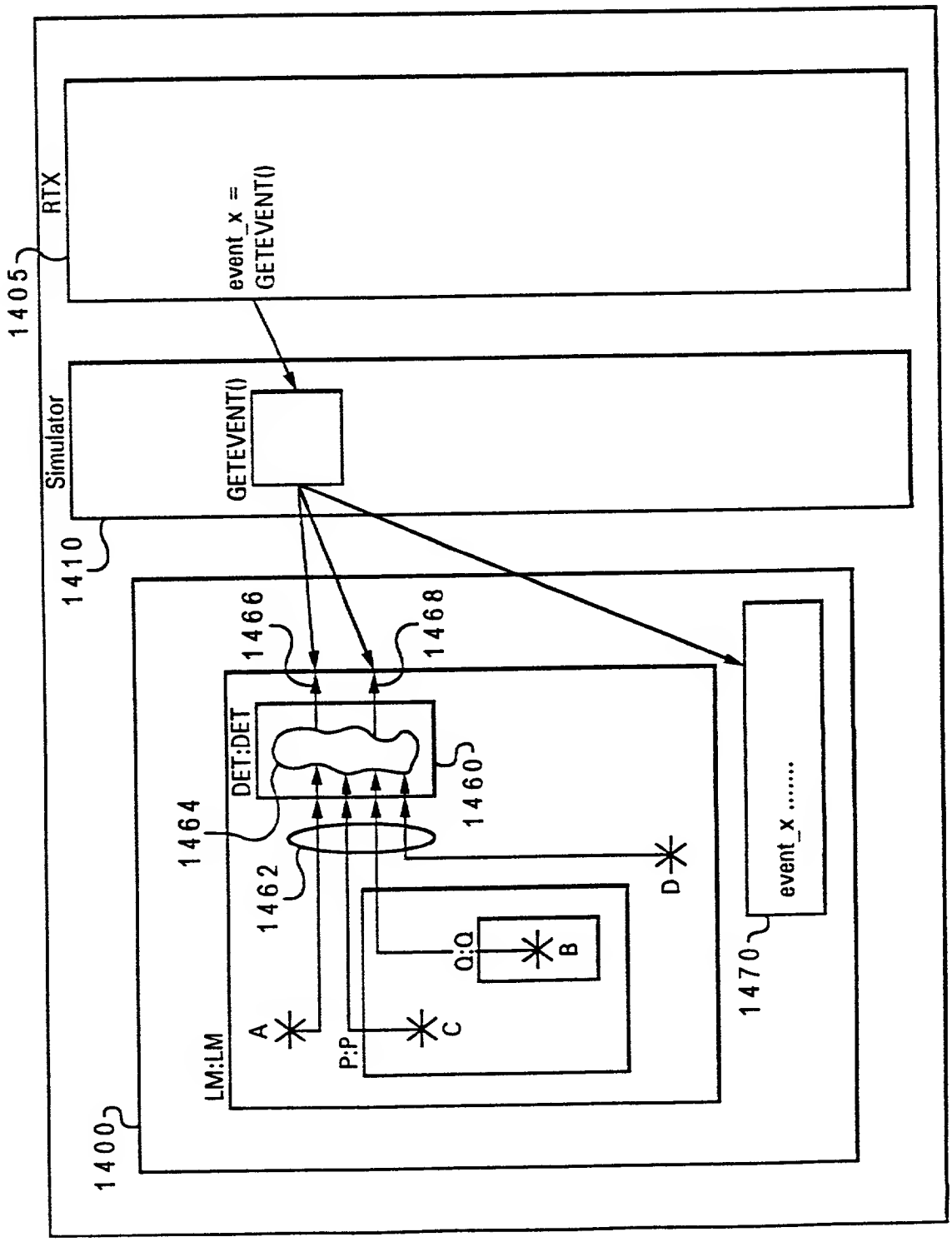

1405
RTX
event_x =
GETEVENT()
Simulator
GETEVENT()
1410
1466
1468
DET:DET
1460
1464
1462
1400
LM:LM
A
P:P
C
Q:Q
B
D
1470
event_x .....

Fig. 14B

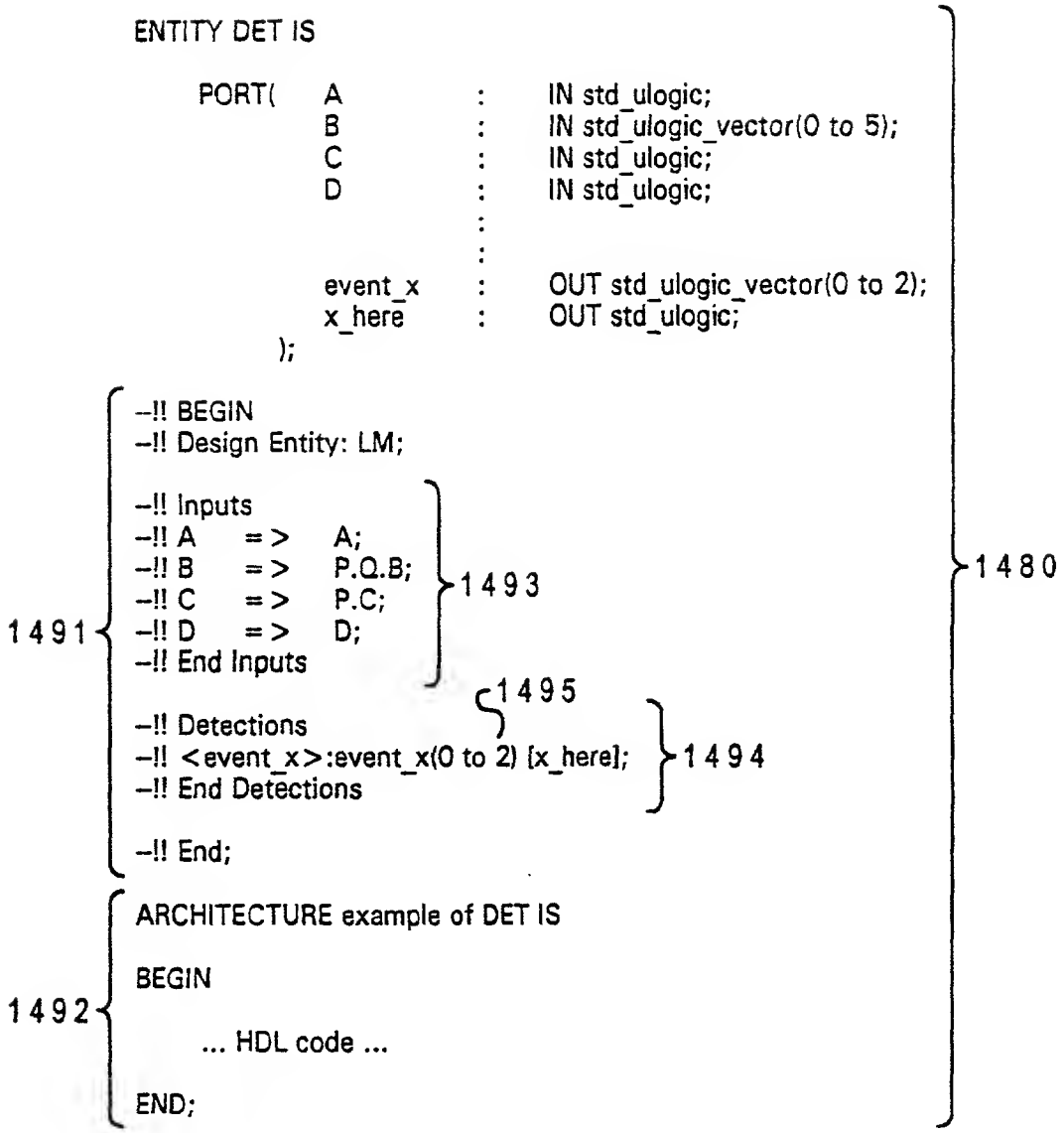

ENTITY DET IS
PORT(A : IN std_ulogic;
B : IN std_ulogic_vector(0 to 5);
C : IN std_ulogic;
D : IN std_ulogic;
event_x : OUT std_ulogic_vector(0 to 2);
x_here : OUT std_ulogic;
);
--!! BEGIN
--!! Design Entity: LM;
--!! Inputs
--!! A => A;
--!! B => P.Q.B;
--!! C => P.C;
--!! D => D;
--!! End Inputs
--!! Detections
--!! <event_x>:event_x(0 to 2) [x_here];
--!! End Detections
--!! End;
ARCHITECTURE example of DET IS
BEGIN
... HDL code ...
END;
1480
1491
1492
1493
1494
1495

Fig. 14C

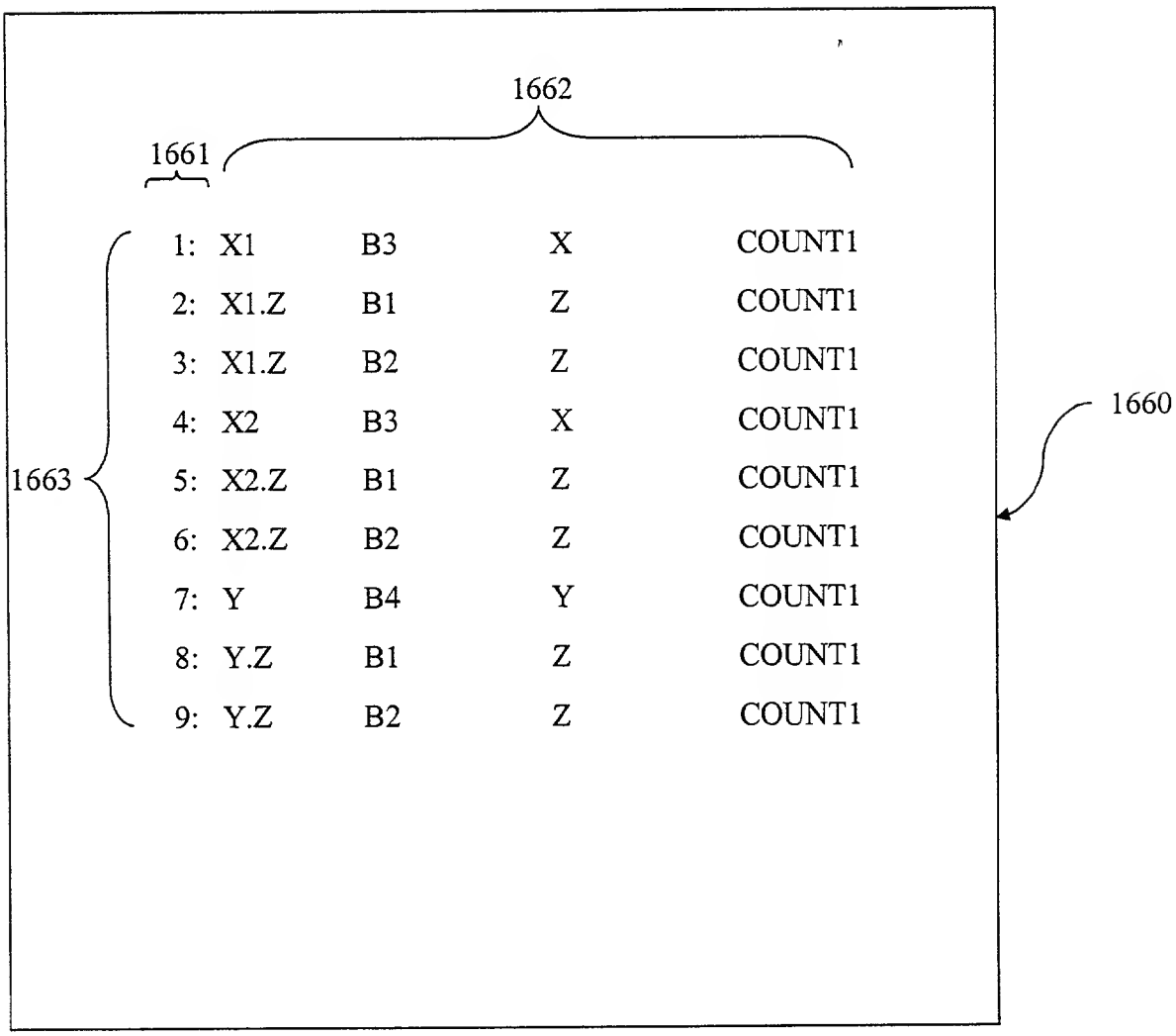

1662
1661
1: X1 B3 X COUNT1
2: X1.Z B1 Z COUNT1
3: X1.Z B2 Z COUNT1
4: X2 B3 X COUNT1
5: X2.Z B1 Z COUNT1
6: X2.Z B2 Z COUNT1
7: Y B4 Y COUNT1
8: Y.Z B1 Z COUNT1
9: Y.Z B2 Z COUNT1
1663
1660

FIG. 15

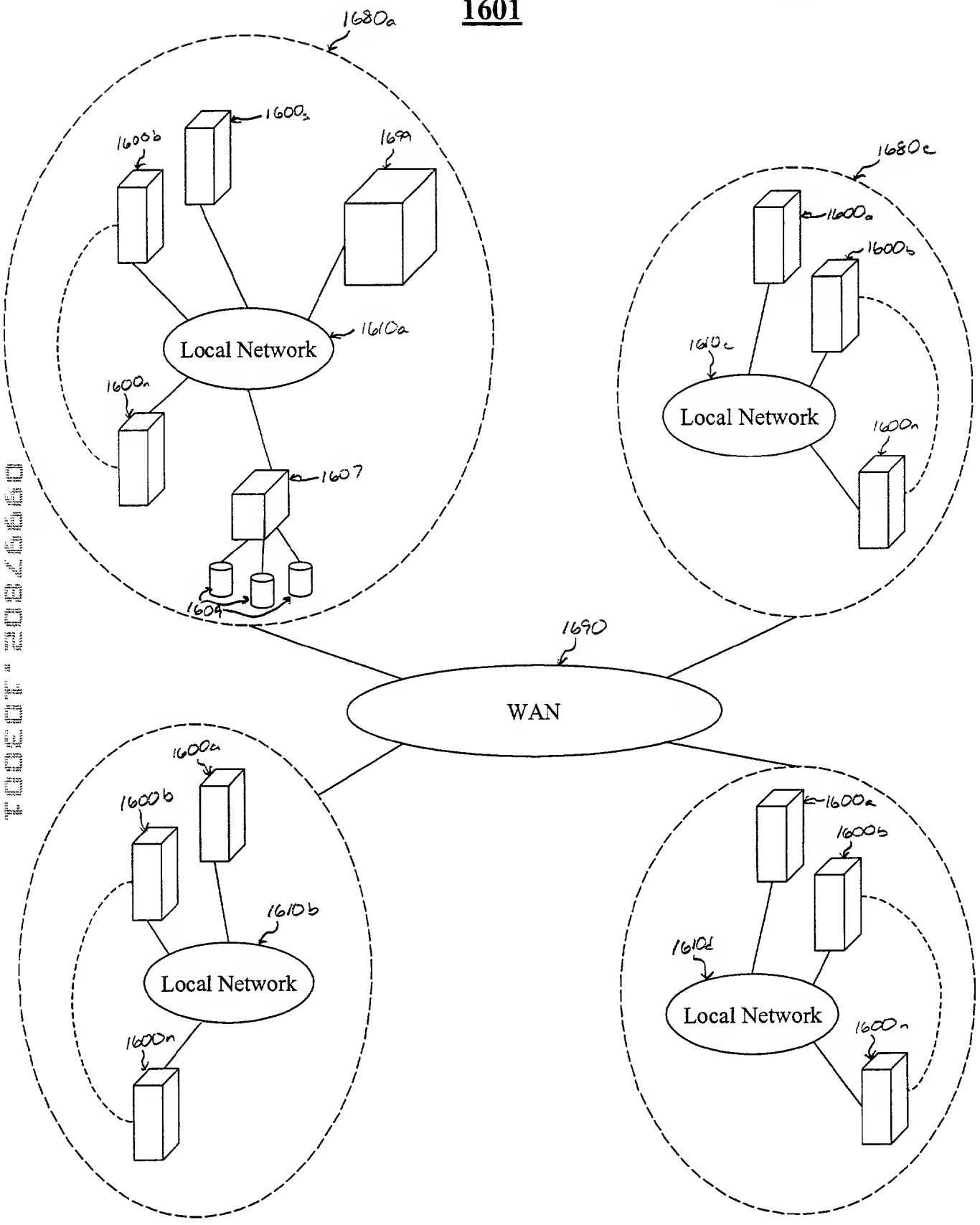

1601
1680a
1600a
1600b
1699
Local Network
1610a
1600n
1607
1690
WAN
1680c
1600a
1600b
1610c
Local Network
1600n
1600a
1600b
1610b
Local Network
1600n
1600a
1600b
1610d
Local Network
1600n

FIG. 16B

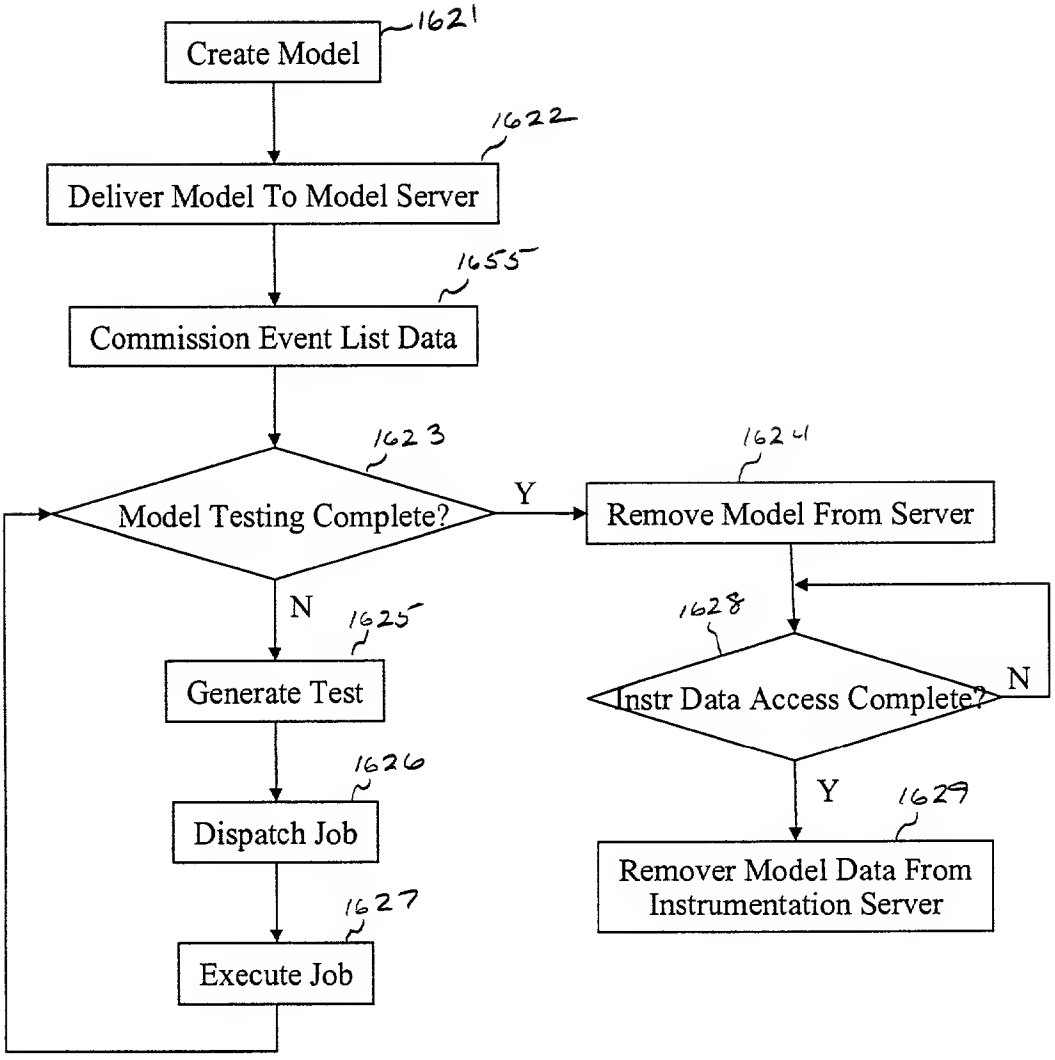

Create Model
1621
Deliver Model To Model Server
1622
Commission Event List Data
1655
Model Testing Complete?
1623
Y
N
Remove Model From Server
1624
Generate Test
1625
Dispatch Job
1626
Execute Job
1627
Instr Data Access Complete?
1628
N
Y
Remover Model Data From Instrumentation Server
1629

FIG. 16C

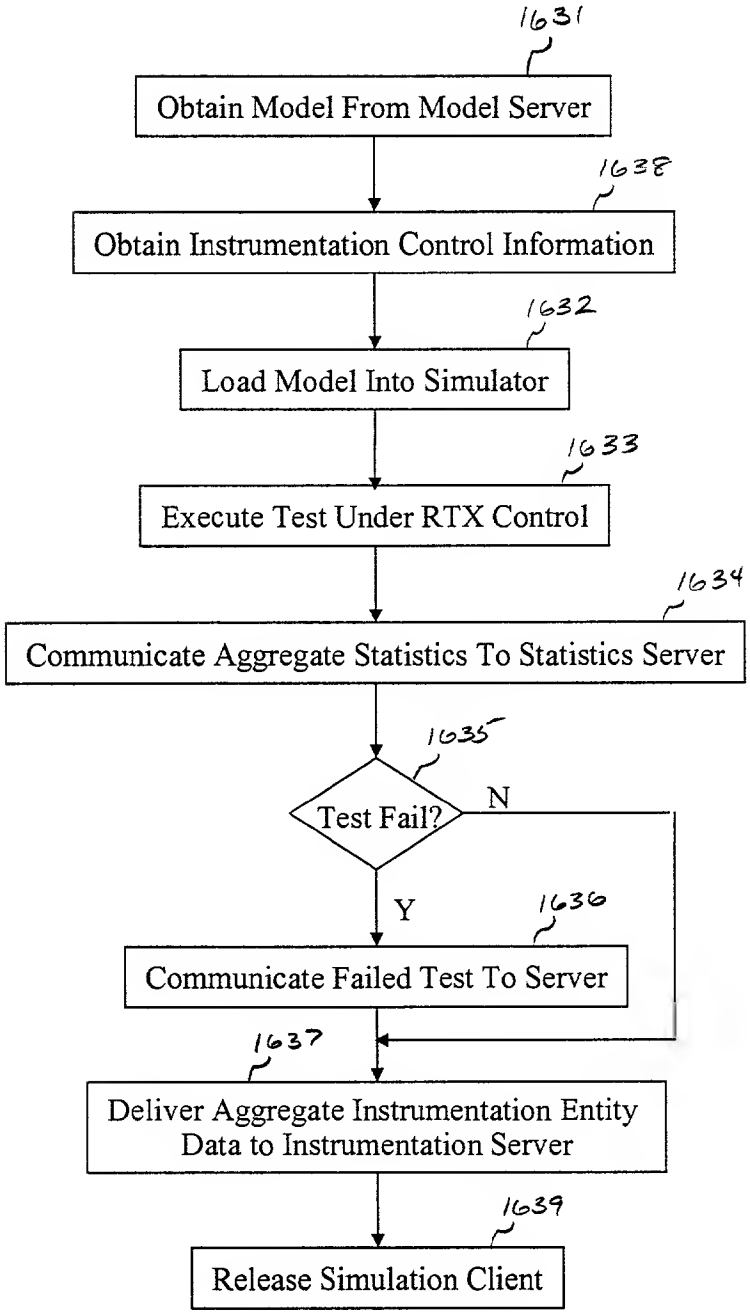

1631
Obtain Model From Model Server
1638
Obtain Instrumentation Control Information
1632
Load Model Into Simulator
1633
Execute Test Under RTX Control
1634
Communicate Aggregate Statistics To Statistics Server
1635
Test Fail?
N
Y
1636
Communicate Failed Test To Server
1637
Deliver Aggregate Instrumentation Entity Data to Instrumentation Server
1639
Release Simulation Client

FIG. 16D

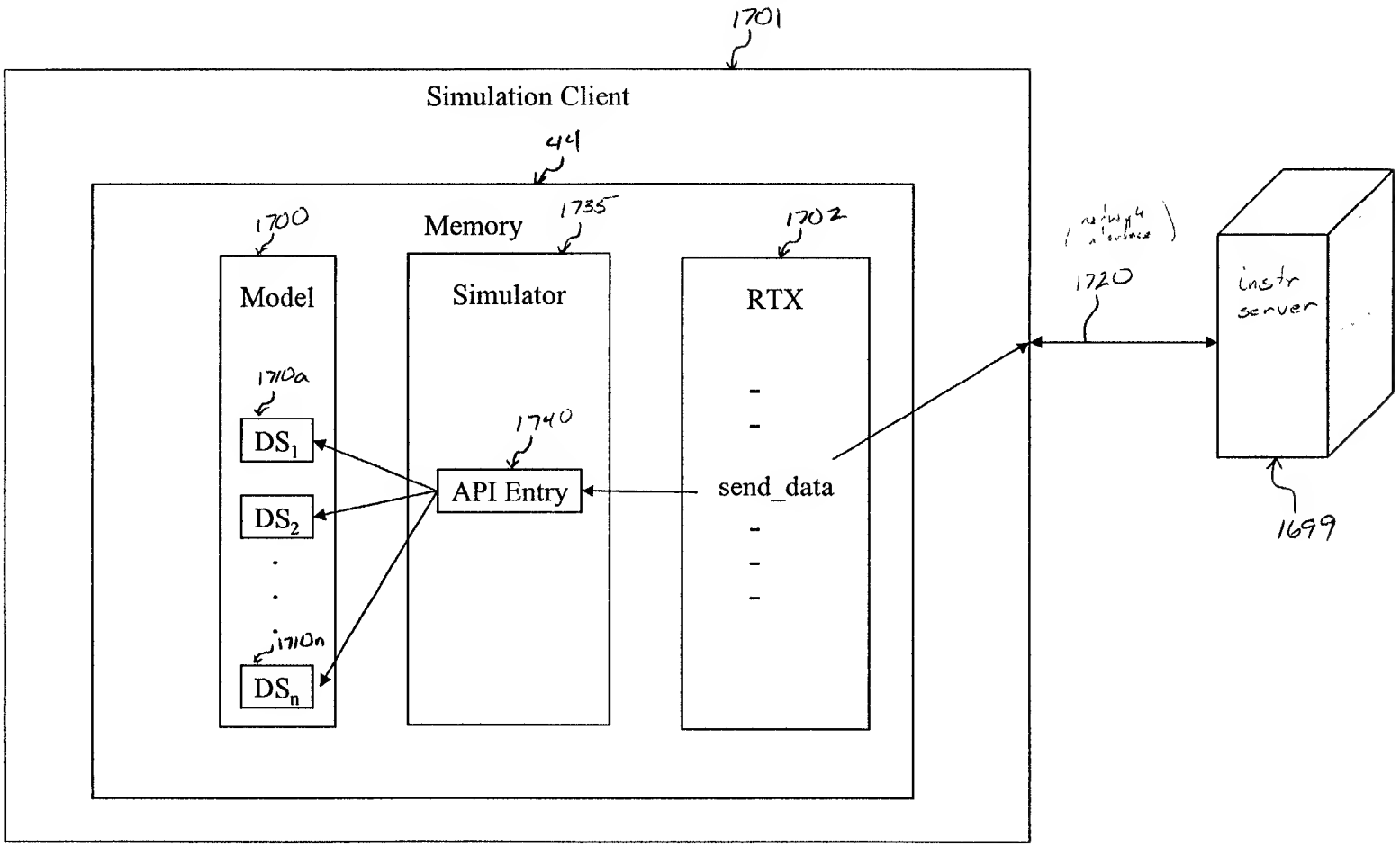

1701
Simulation Client
44
1700
Model
Memory
1735
Simulator
1702
RTX
1710a
DS1
DS2
1710n
DSn
1740
API Entry
send_data
1720
instr server
1699

FIG. 17A

1750

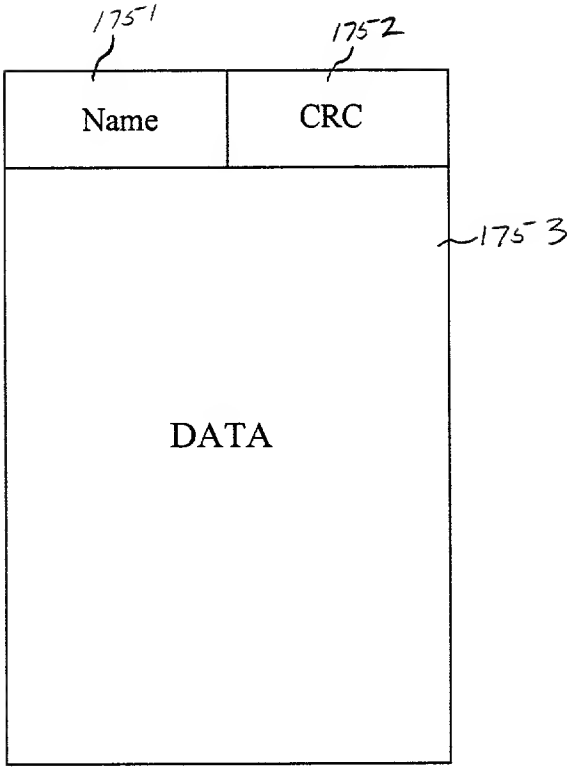


FIG. 17B

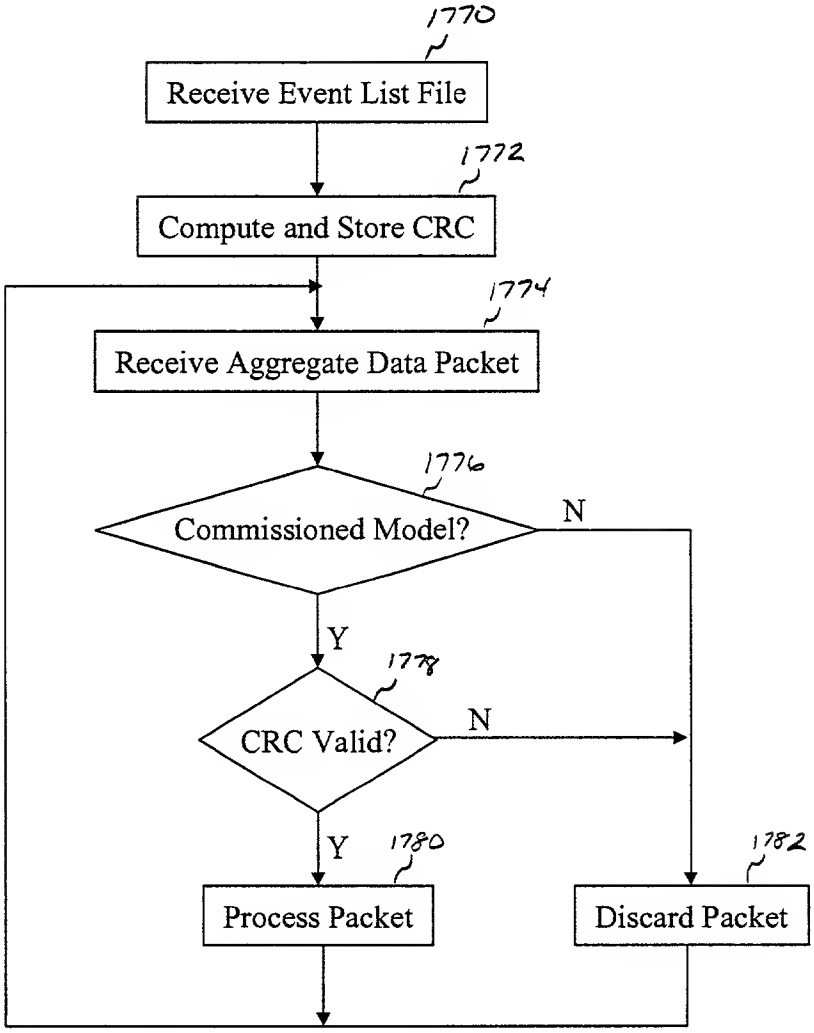

1770
Receive Event List File
1772
Compute and Store CRC
1774
Receive Aggregate Data Packet
1776
Commissioned Model?
N
Y
1778
CRC Valid?
N
Y
1780
Process Packet
1782
Discard Packet

FIG. 17C

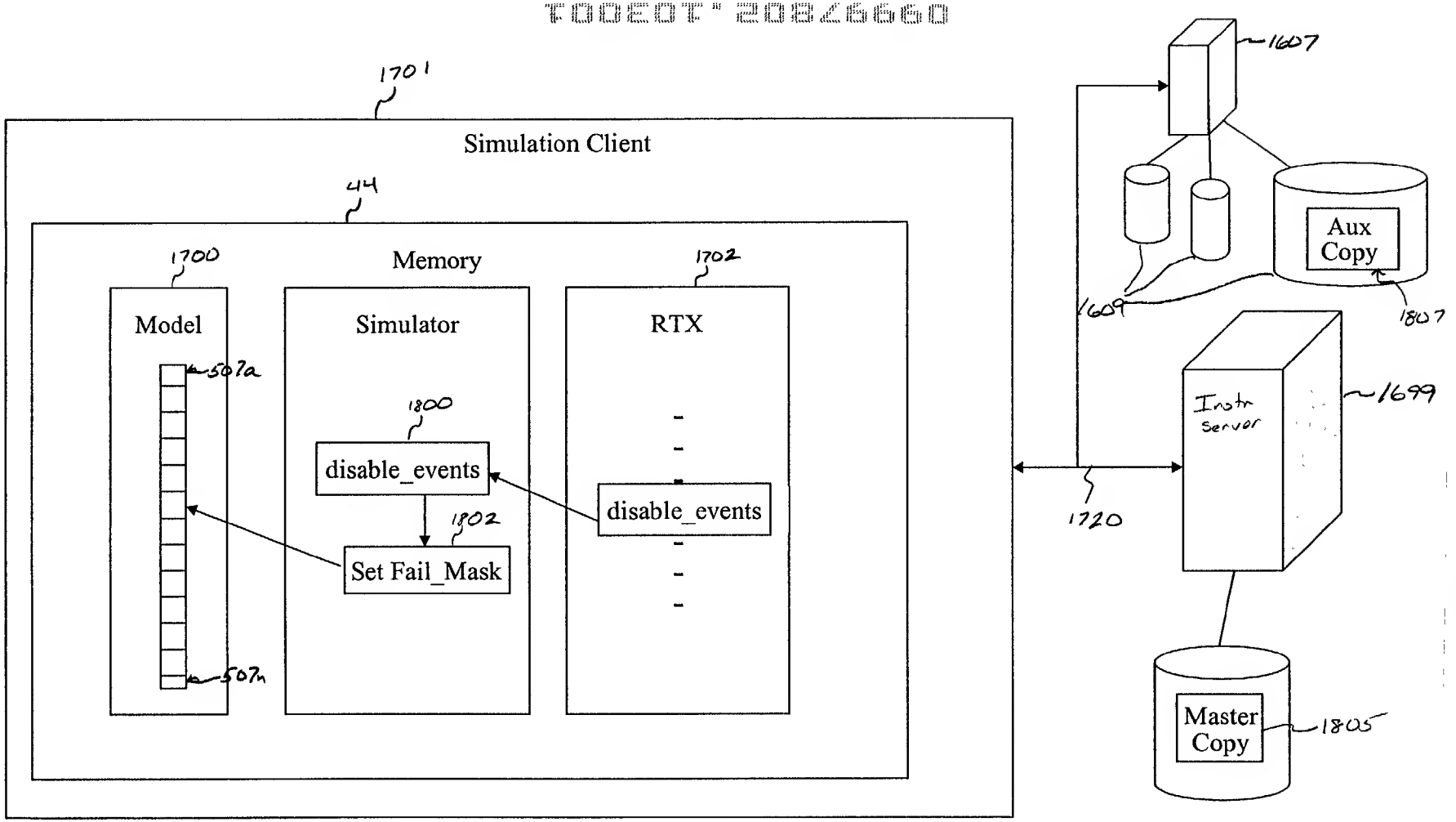

1701
Simulation Client
44
1700
Memory
1702
Model
Simulator
RTX
507a
507n
1800
disable_events
1802
Set Fail_Mask
disable_events
1607
Aux Copy
1609
1807
1699
Instr Server
1720
Master Copy
1805

FIG. 18A

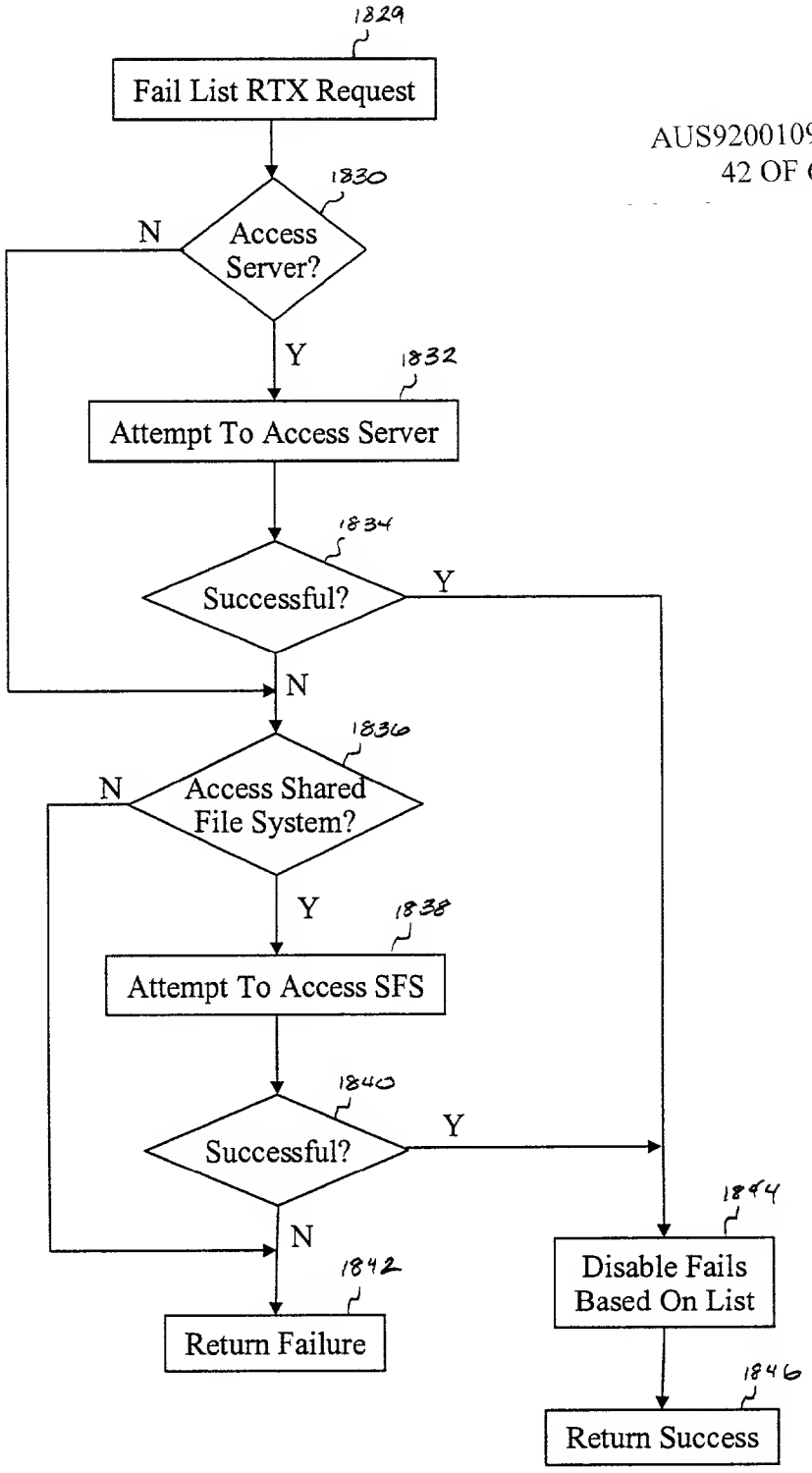

1829
Fail List RTX Request
1830
Access Server?
N
Y
1832
Attempt To Access Server
1834
Successful?
Y
N
1836
Access Shared File System?
N
Y
1838
Attempt To Access SFS
1840
Successful?
Y
N
1842
Return Failure
1844
Disable Fails Based On List
1846
Return Success

FIG. 18B

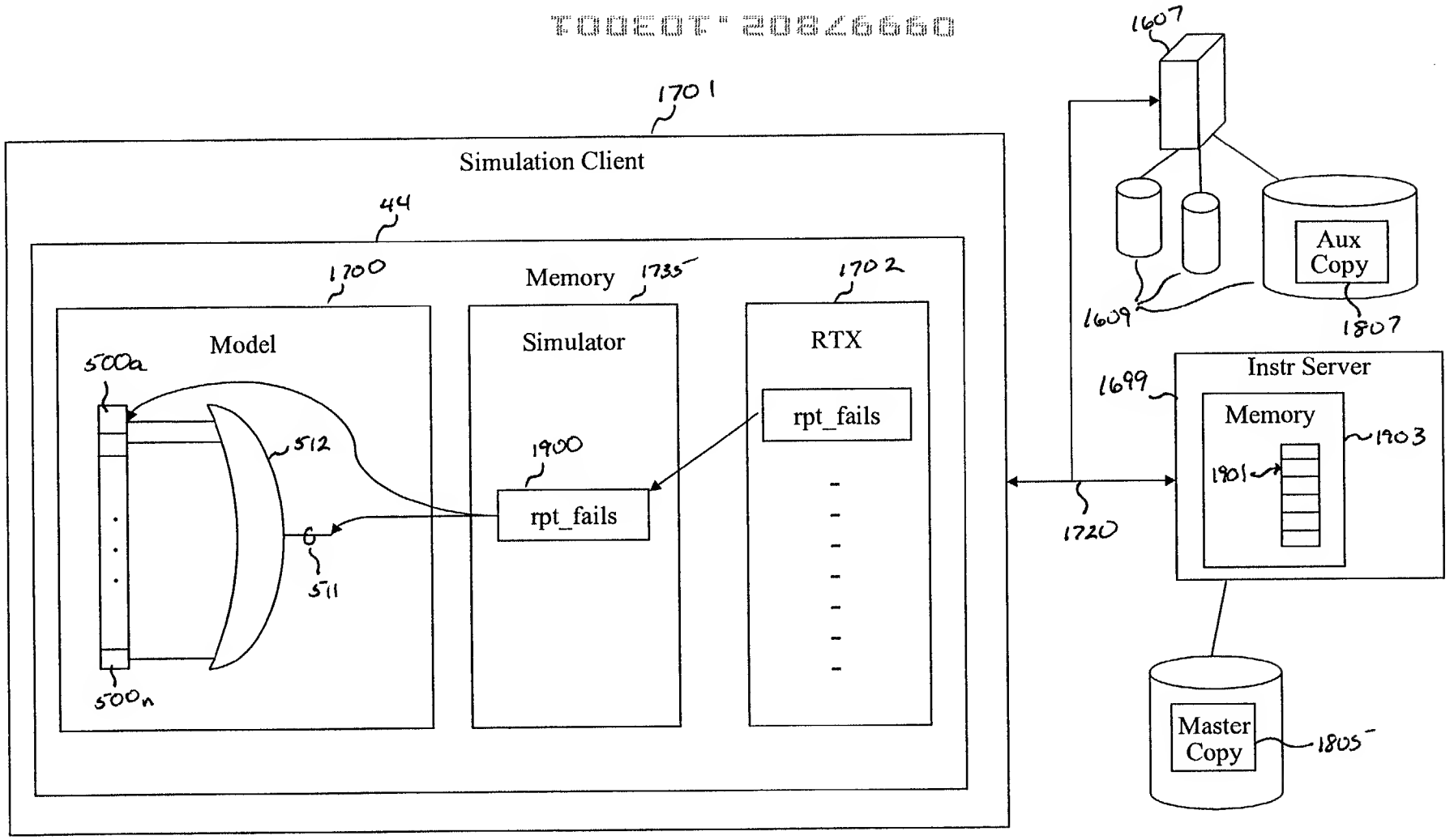

1607
1701
Simulation Client
44
1700
Memory
1735
1702
Model
Simulator
RTX
5002
512
1900
rpt_fails
rpt_fails
511
5001
1609
Aux Copy
1807
1699
Instr Server
Memory
1903
1901
1720
Master Copy
1805

FIG. 19A

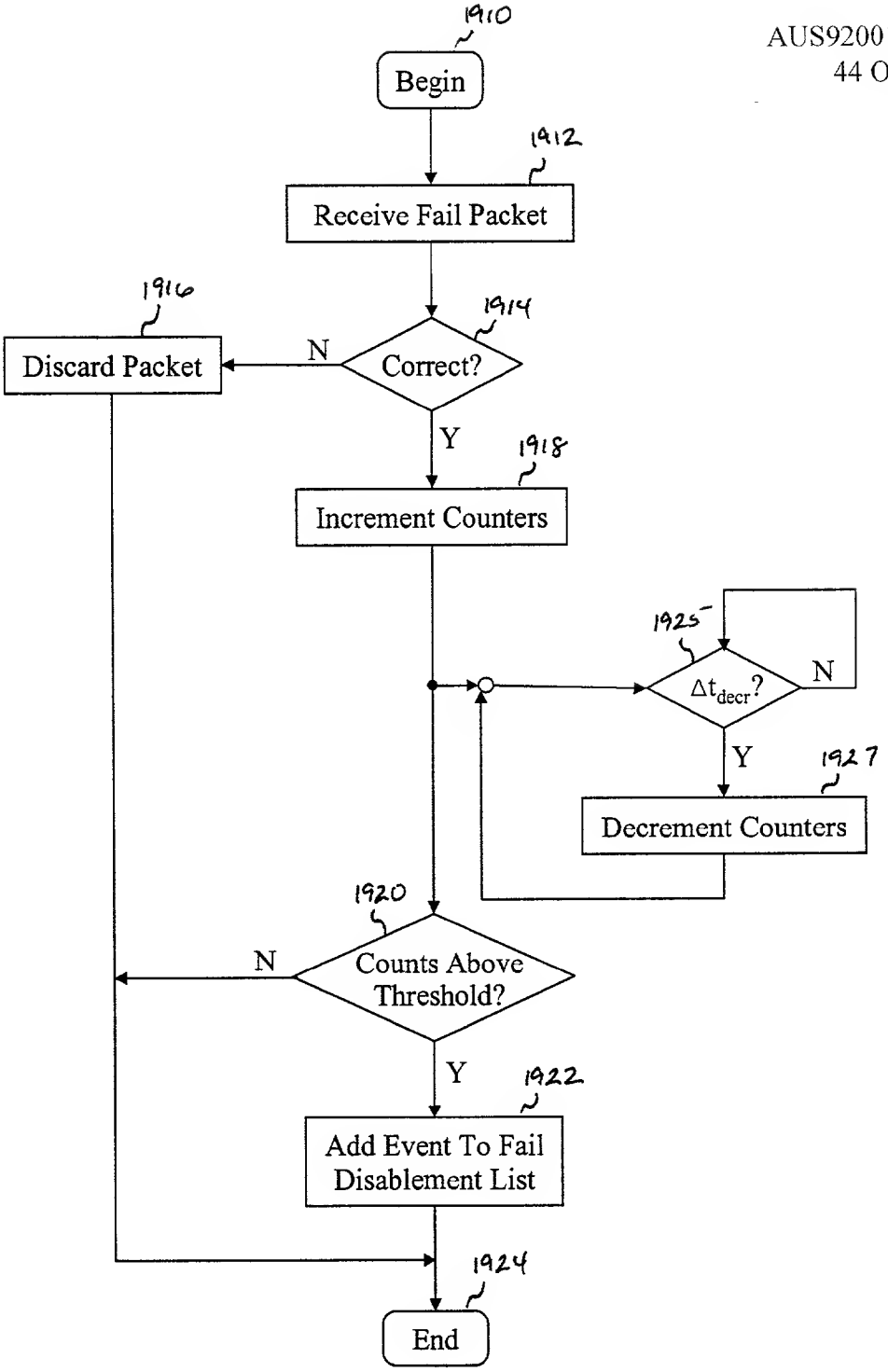

1910
Begin
1912
Receive Fail Packet
1914
Correct?
N
1916
Discard Packet
Y
1918
Increment Counters
1925
Δt_{decr}?
N
Y
1927
Decrement Counters
1920
Counts Above Threshold?
N
Y
1922
Add Event To Fail Disablement List
1924
End

FIG. 19B

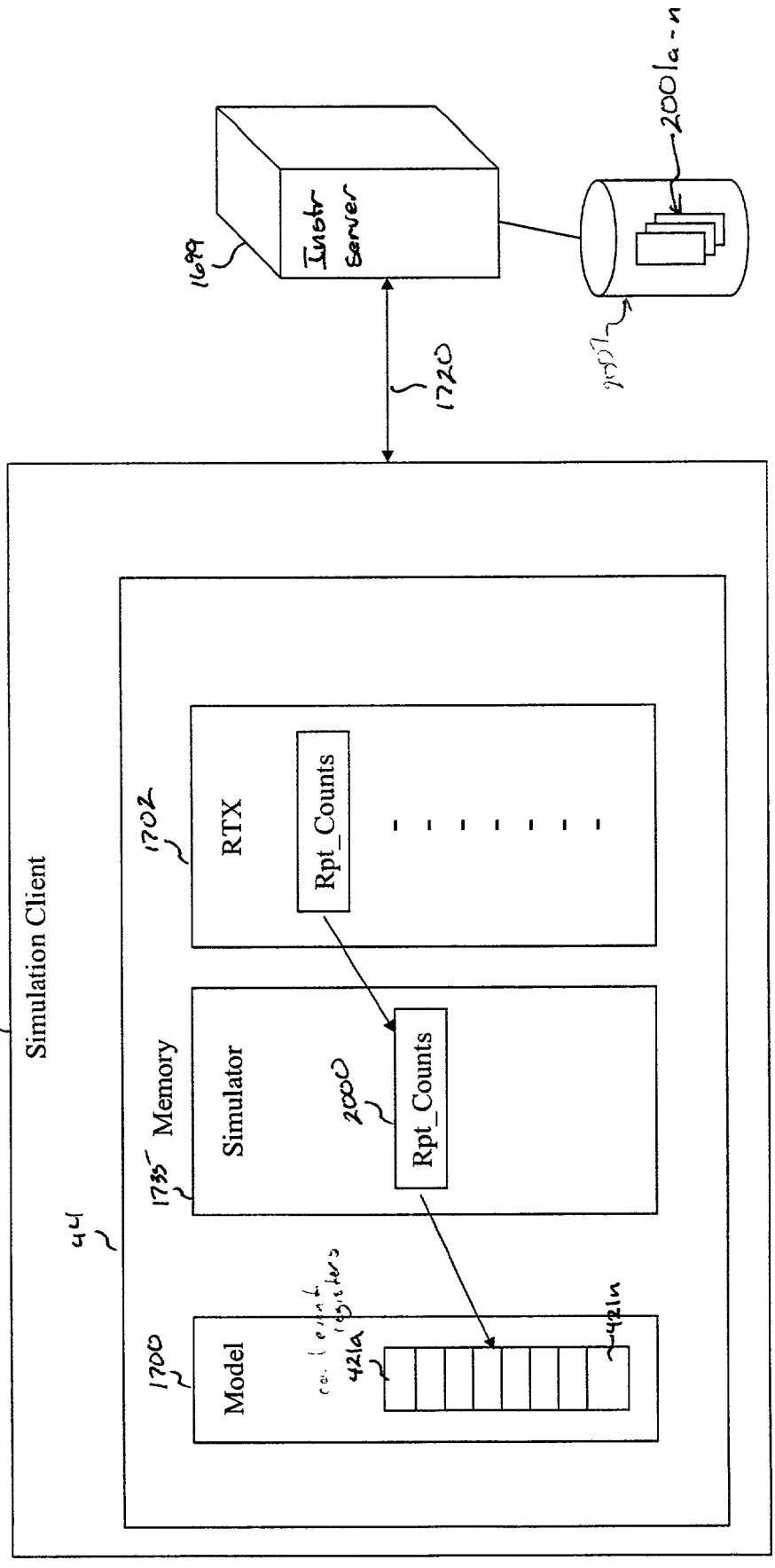

1701
Simulation Client
44
1735
Memory
Simulator
2000
Rpt_Counts
1702
RTX
Rpt_Counts
1700
Model
421a
421n
1699
Instr Server
1720
2001a - n

FIG. 20A

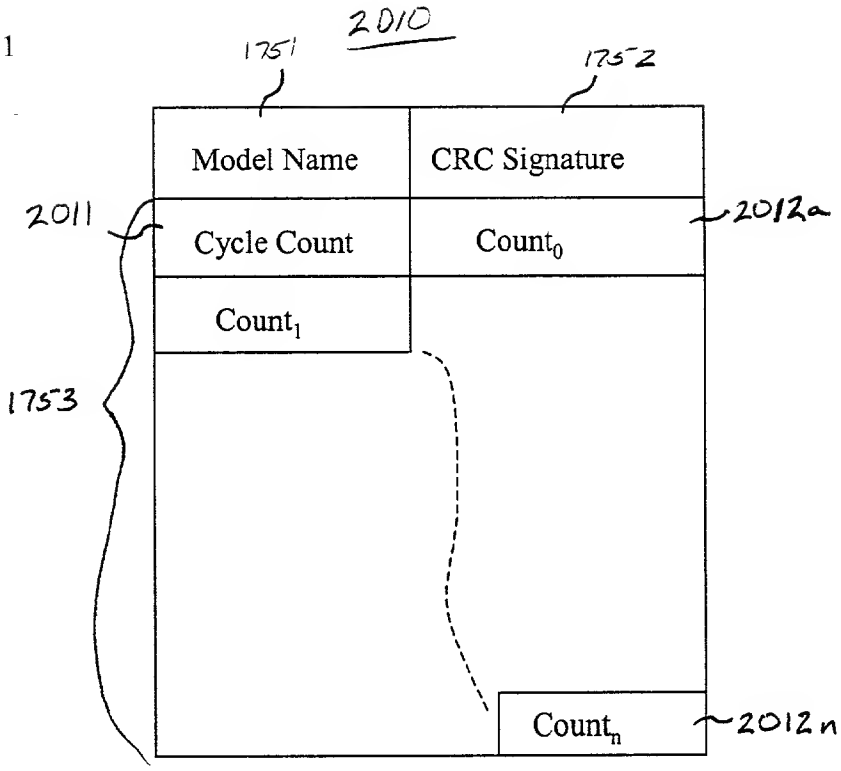


FIG. 20B

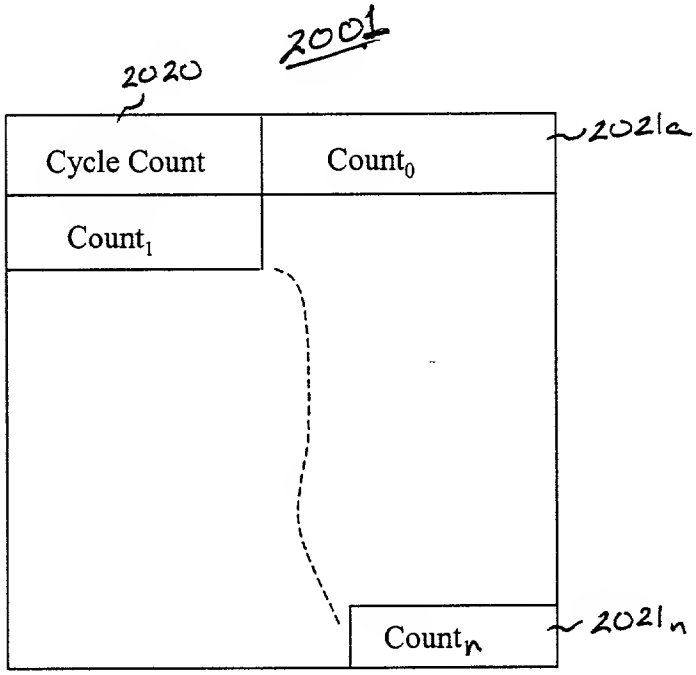


FIG. 20C

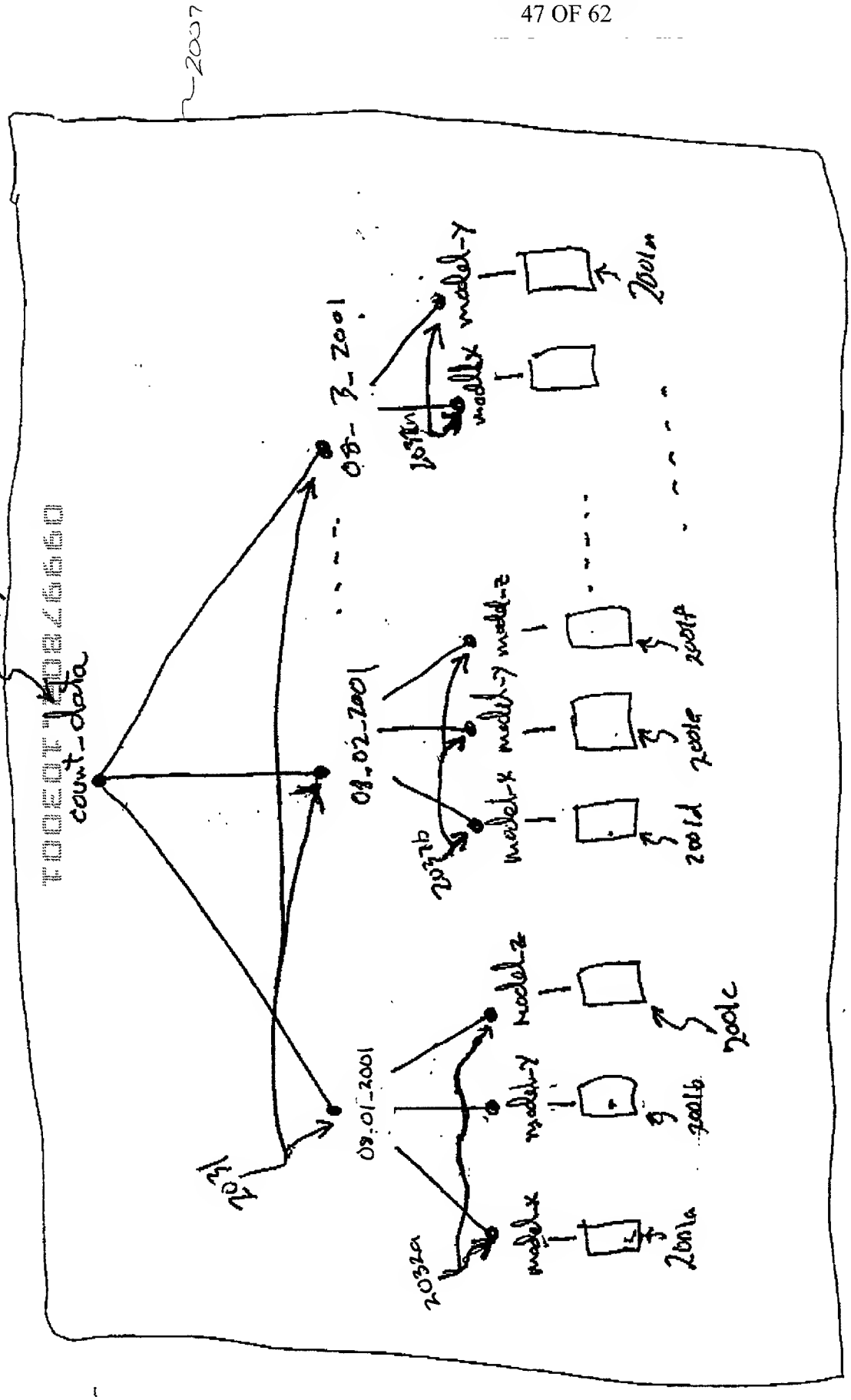

2070
count_data
2074
09_01_2001
model_x
model_y
model_z
2076a
2078a
2080a
09_02_2001
2076b
2078b
2080b
09_03_2001
2076c
2078c
2080c

FIG. 20D

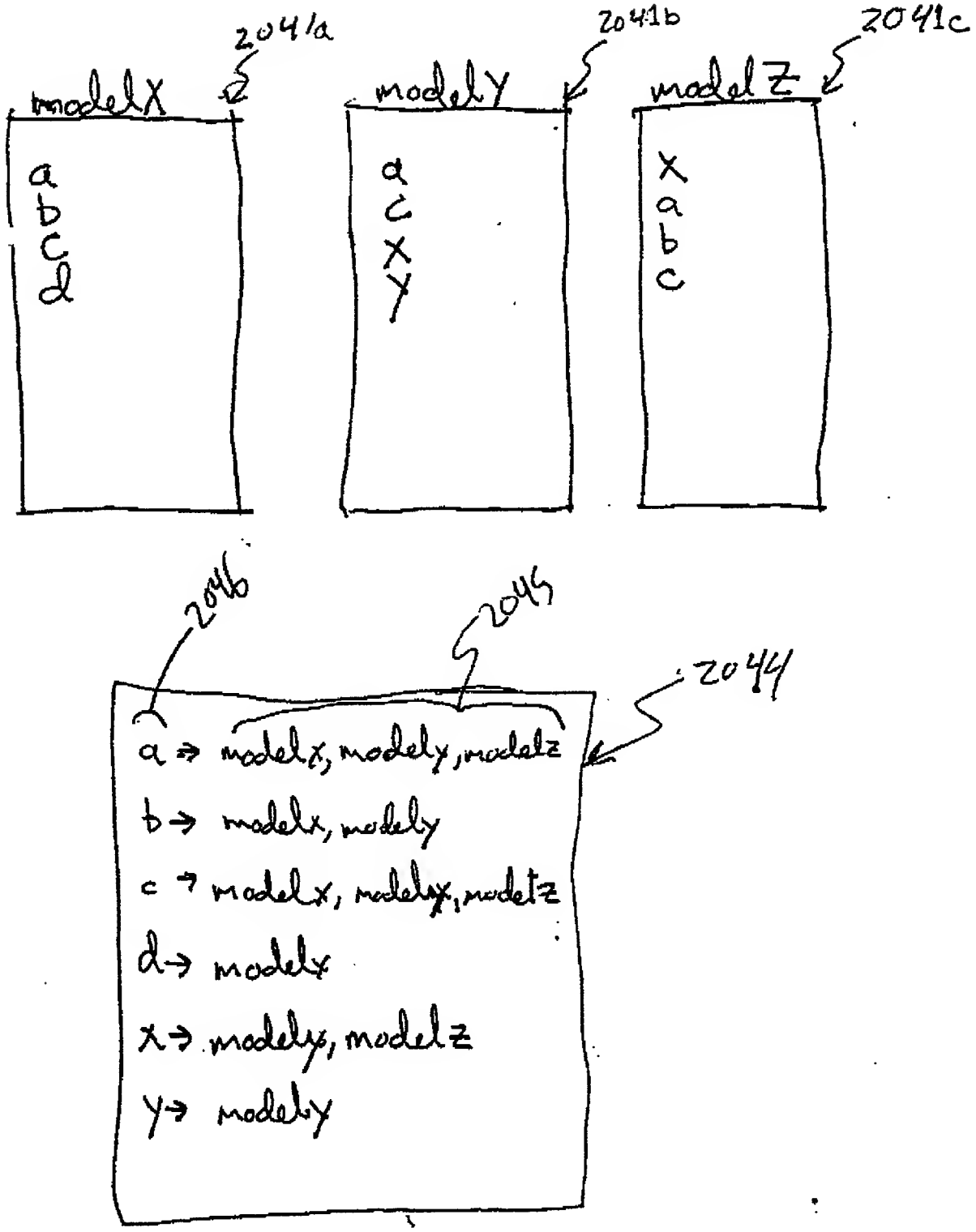


FIG. 20E

2052
Counter Query Manager
2059
CQE
44
Command entity list
2053a
Counter queries
2053n
2030
Counter Sharing Directory
2031a
2031q
2055
Basic Counter Output Report Subdirectory
2061a
2061b
2061n
2044z
Translation Table
a→....
b→....
1720
2056
1600
Query Interface
2040
2041
2007

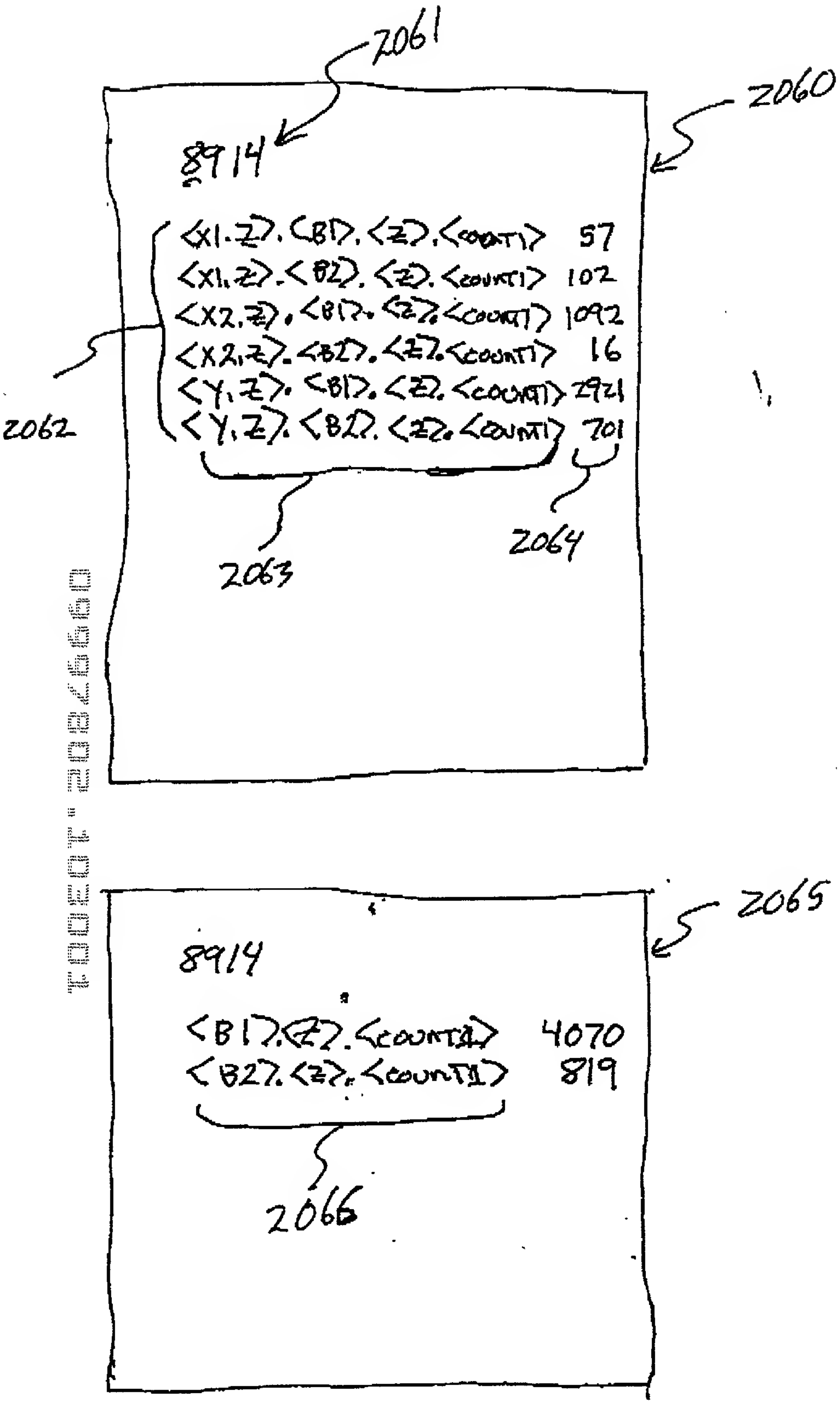


FIG. 206

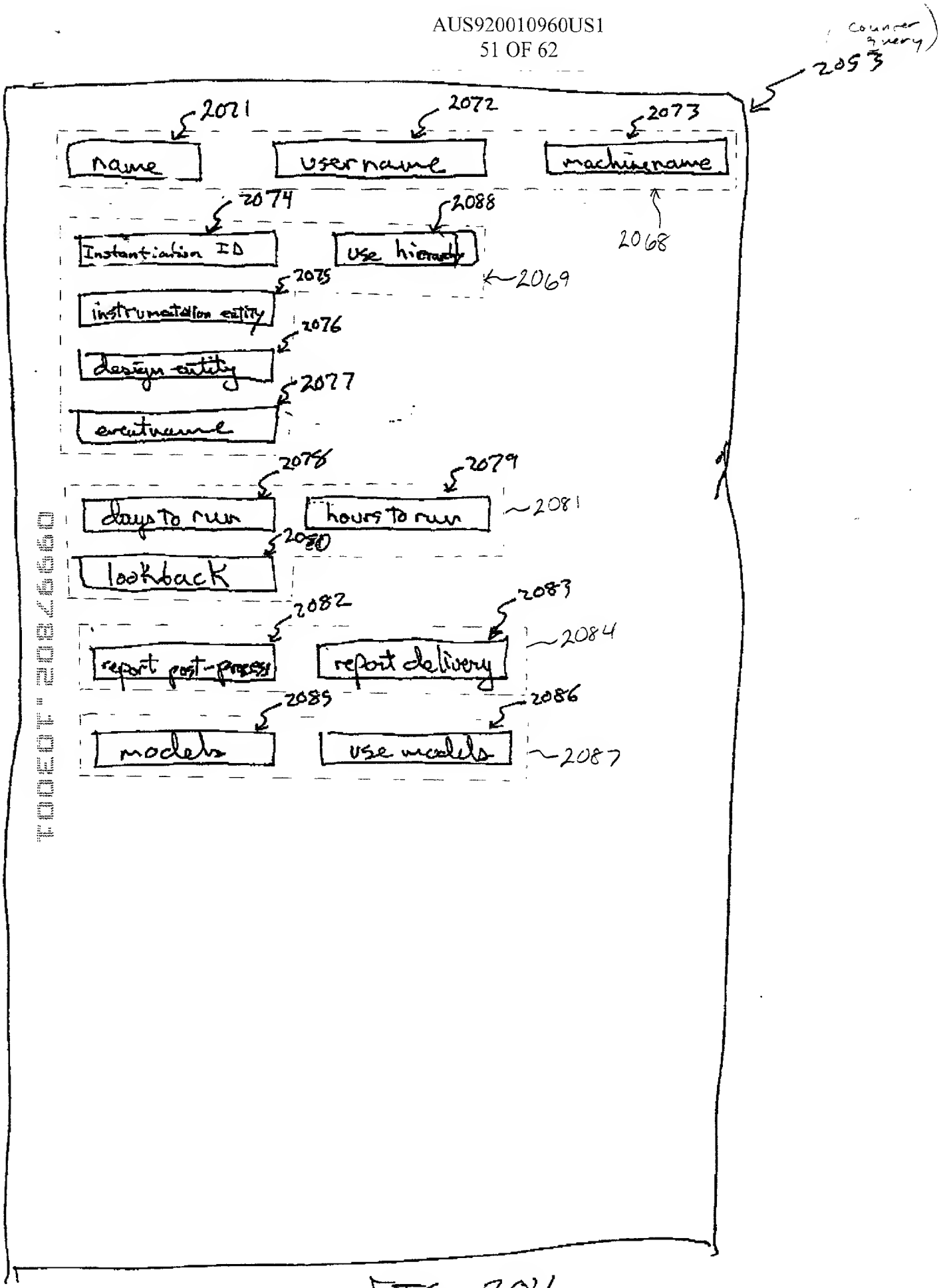


FIG. 20H

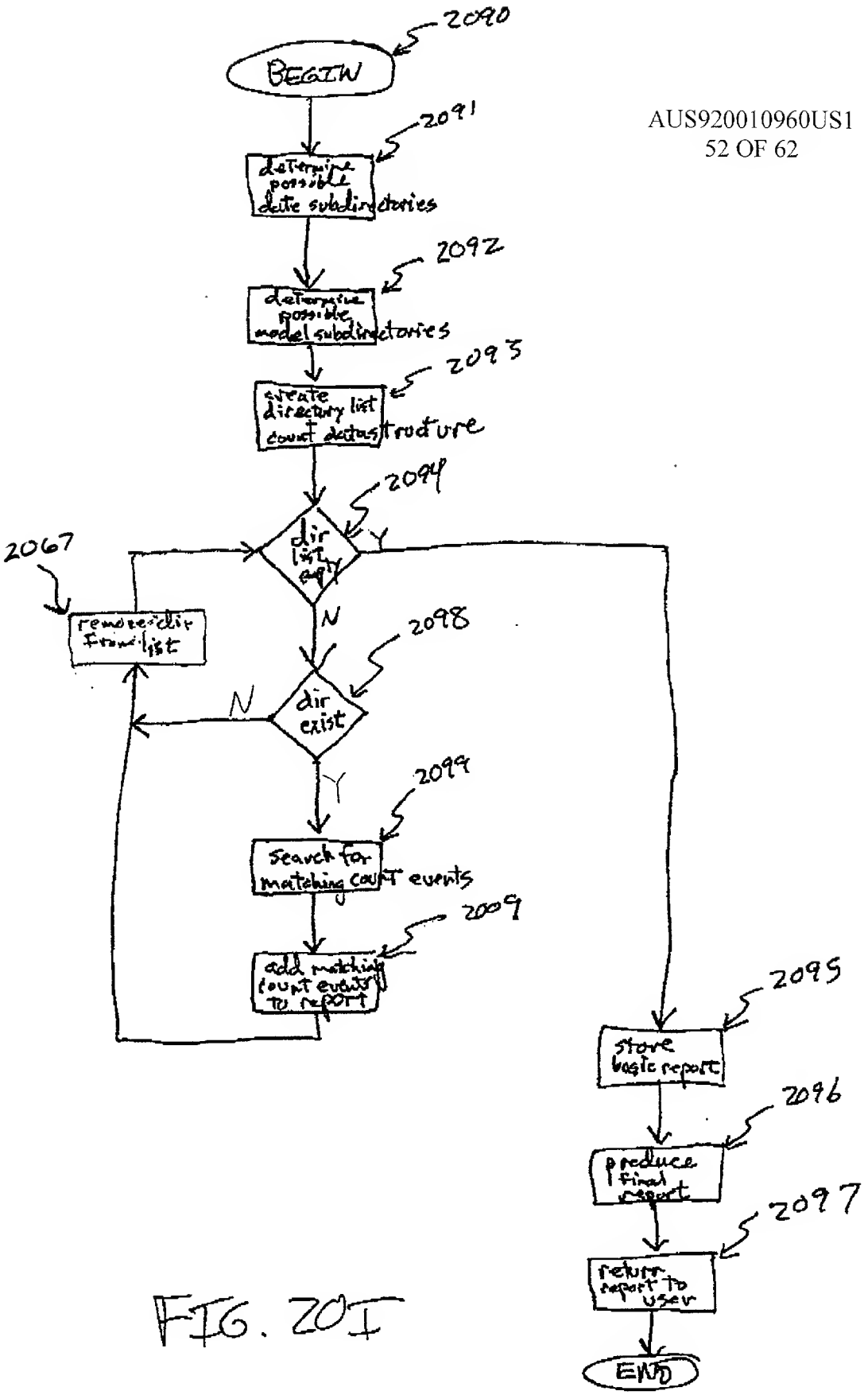


FIG. 20I

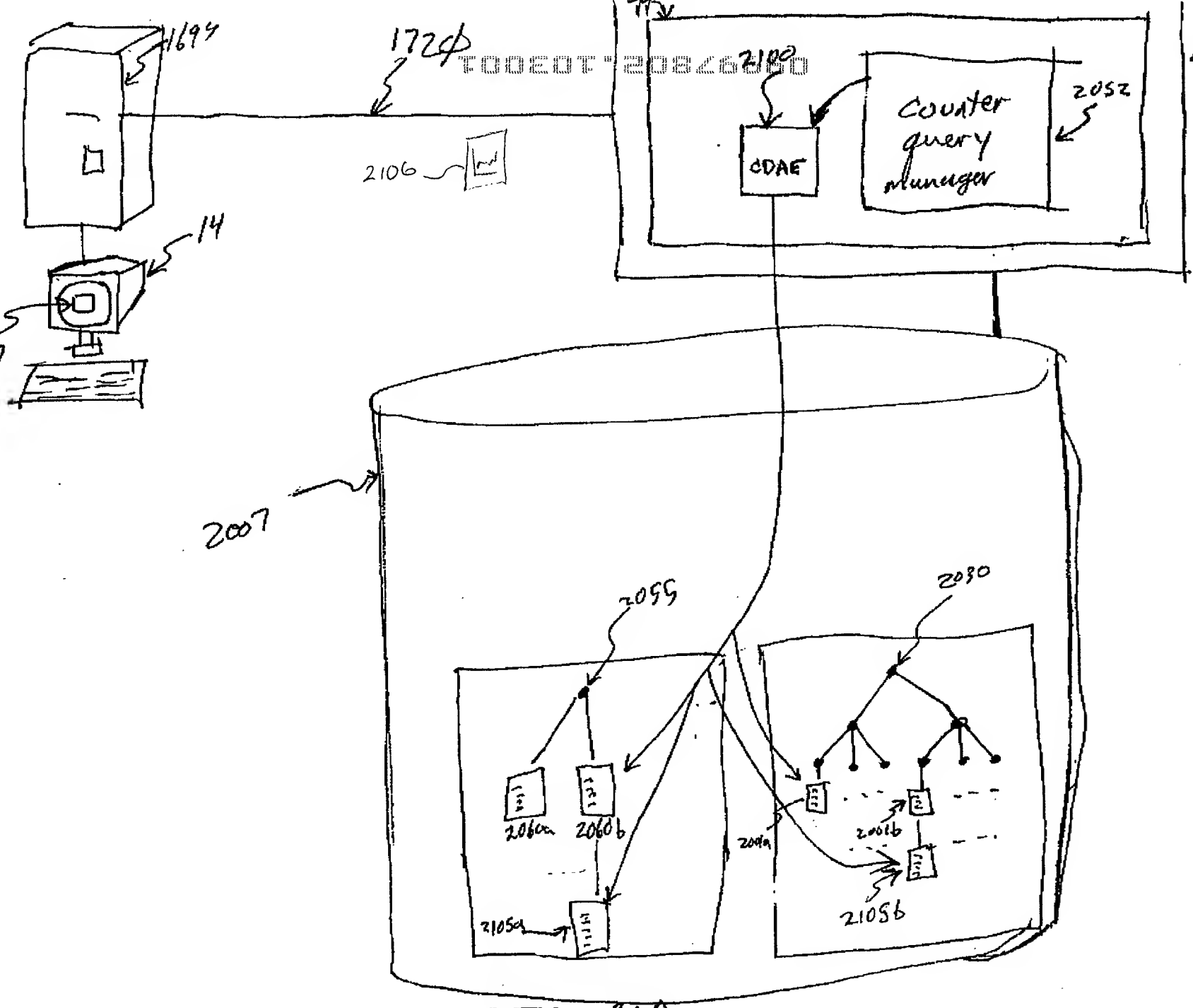

1694
1720
2100
2052
Counter query manager
CDAE
2106
14
2107
2007
2055
2030
2060a
2060b
2105a
2105b
2001a
2001b

FIG. 21A

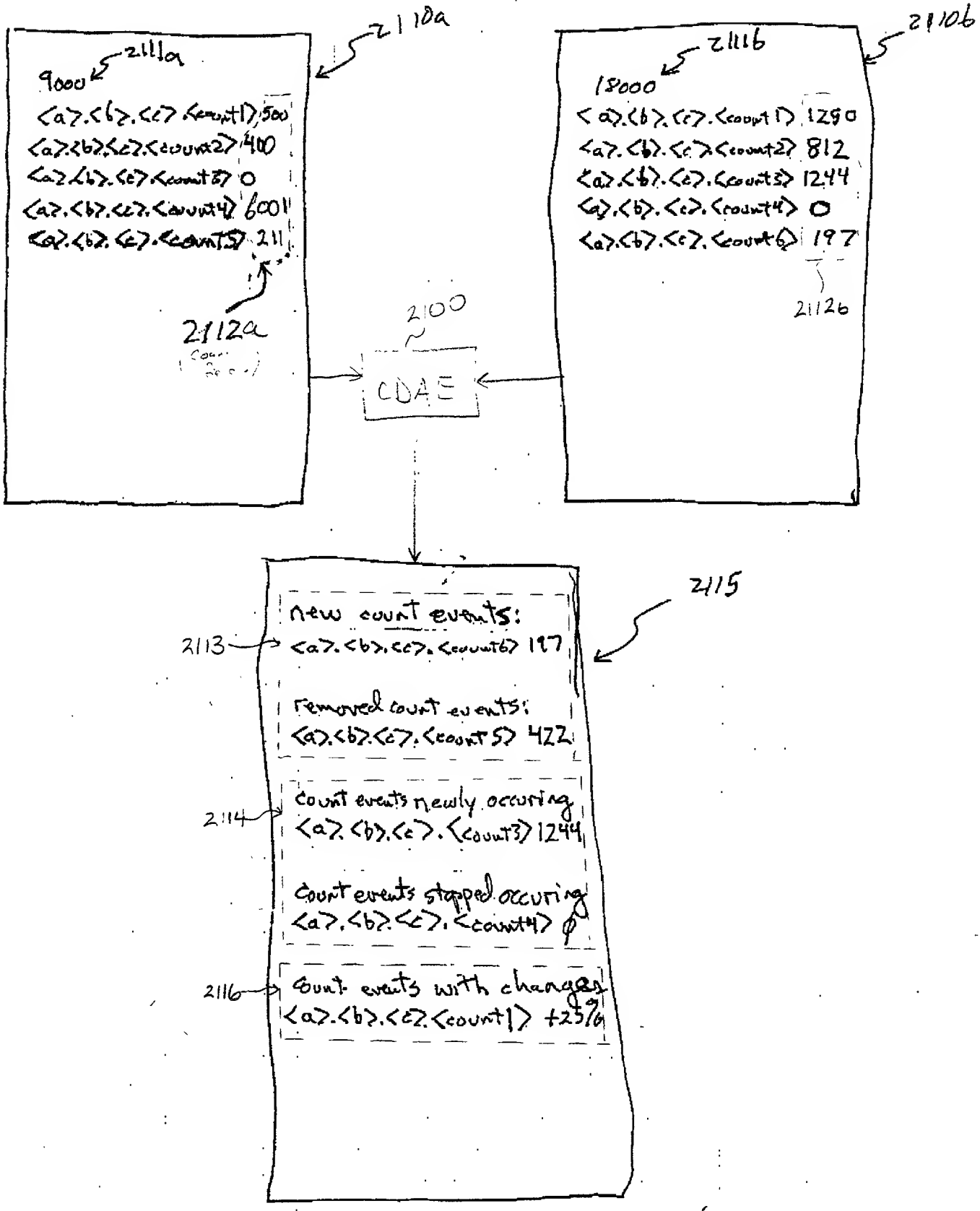

2110a
2111a
9000
<a>.<b>.<c>.<count1> 500
<a>.<b>.<c>.<count2> 400
<a>.<b>.<c>.<count3> 0
<a>.<b>.<c>.<count4> 6001
<a>.<b>.<c>.<count5> 211
2112a
2110b
2111b
18000
<a>.<b>.<c>.<count1> 1290
<a>.<b>.<c>.<count2> 812
<a>.<b>.<c>.<count3> 1244
<a>.<b>.<c>.<count4> 0
<a>.<b>.<c>.<count6> 197
2112b
2100
CDAE
2115
2113
new count events:
<a>.<b>.<c>.<count6> 197
removed count events:
<a>.<b>.<c>.<count5> 422
2114
count events newly occurring
<a>.<b>.<c>.<count3> 1244
count events stopped occurring
<a>.<b>.<c>.<count4> ϕ
2116
count events with changes
<a>.<b>.<c>.<count1> +25%

FIG. 21B

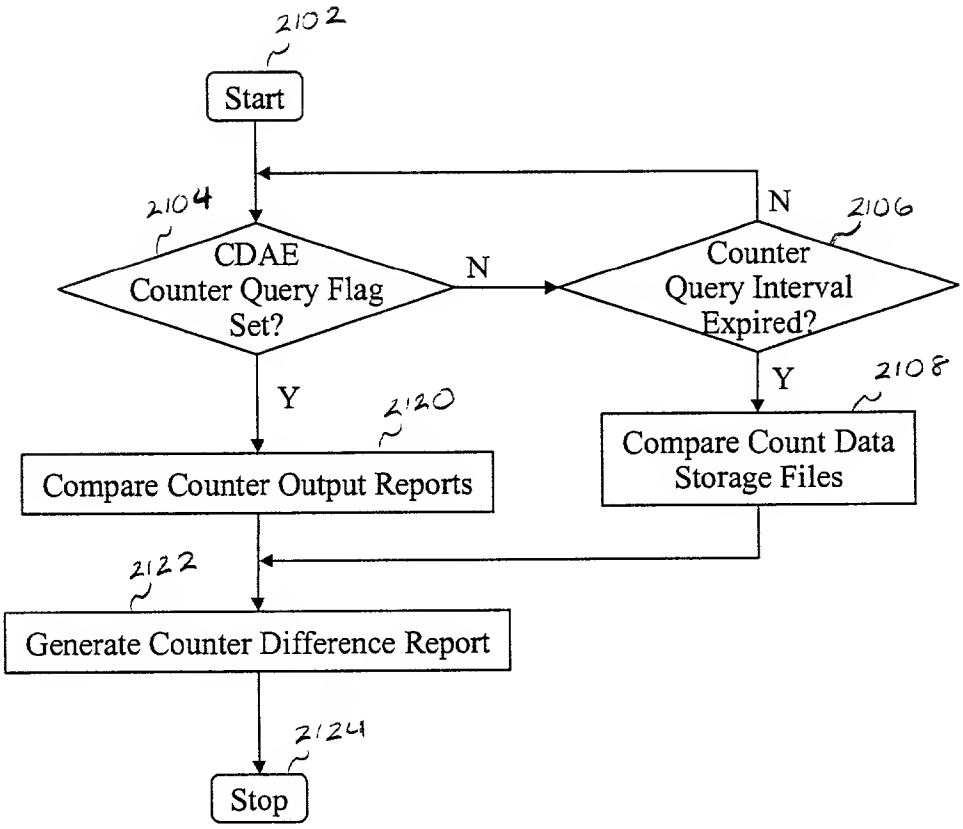


FIG. 21C

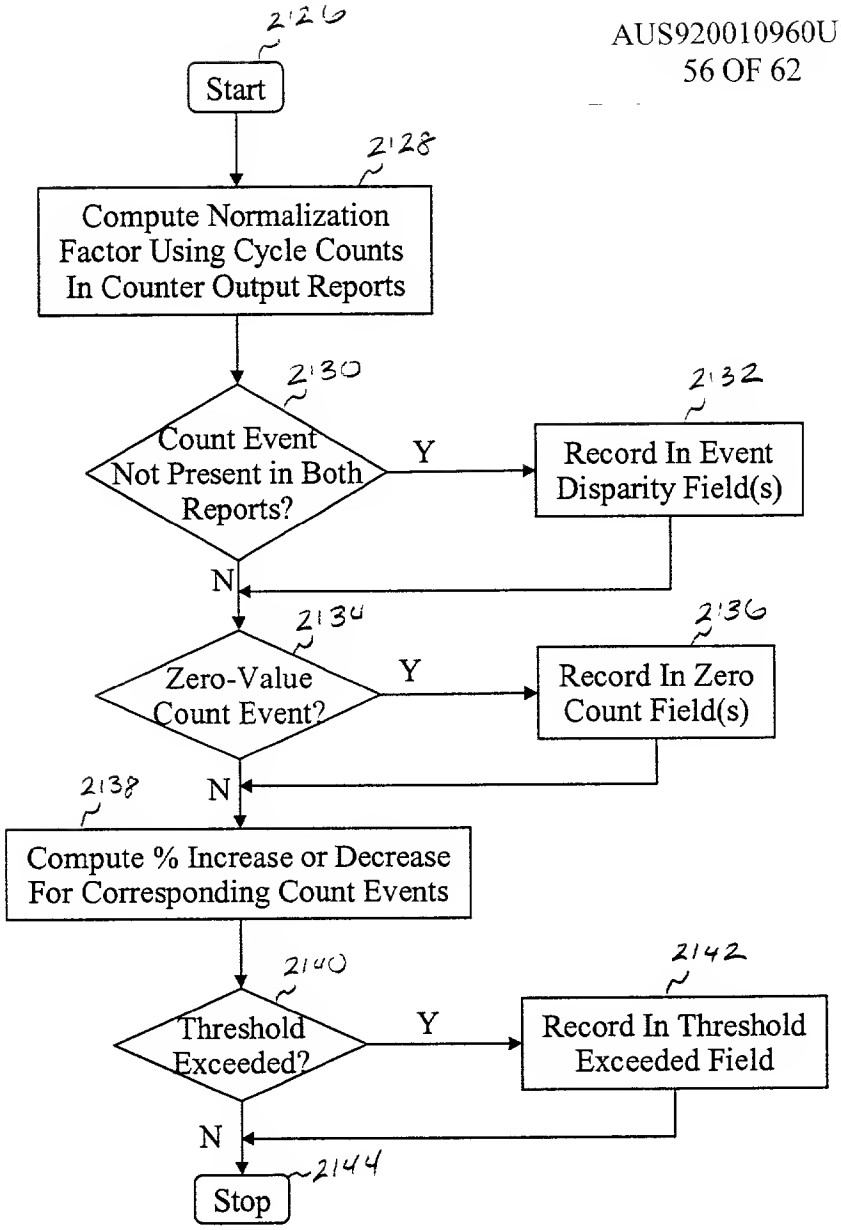

2126
Start
2128
Compute Normalization Factor Using Cycle Counts In Counter Output Reports
2130
Count Event Not Present in Both Reports?
Y
2132
Record In Event Disparity Field(s)
N
2134
Zero-Value Count Event?
Y
2136
Record In Zero Count Field(s)
N
2138
Compute % Increase or Decrease For Corresponding Count Events
2140
Threshold Exceeded?
Y
2142
Record In Threshold Exceeded Field
N
2144
Stop

FIG. 21D

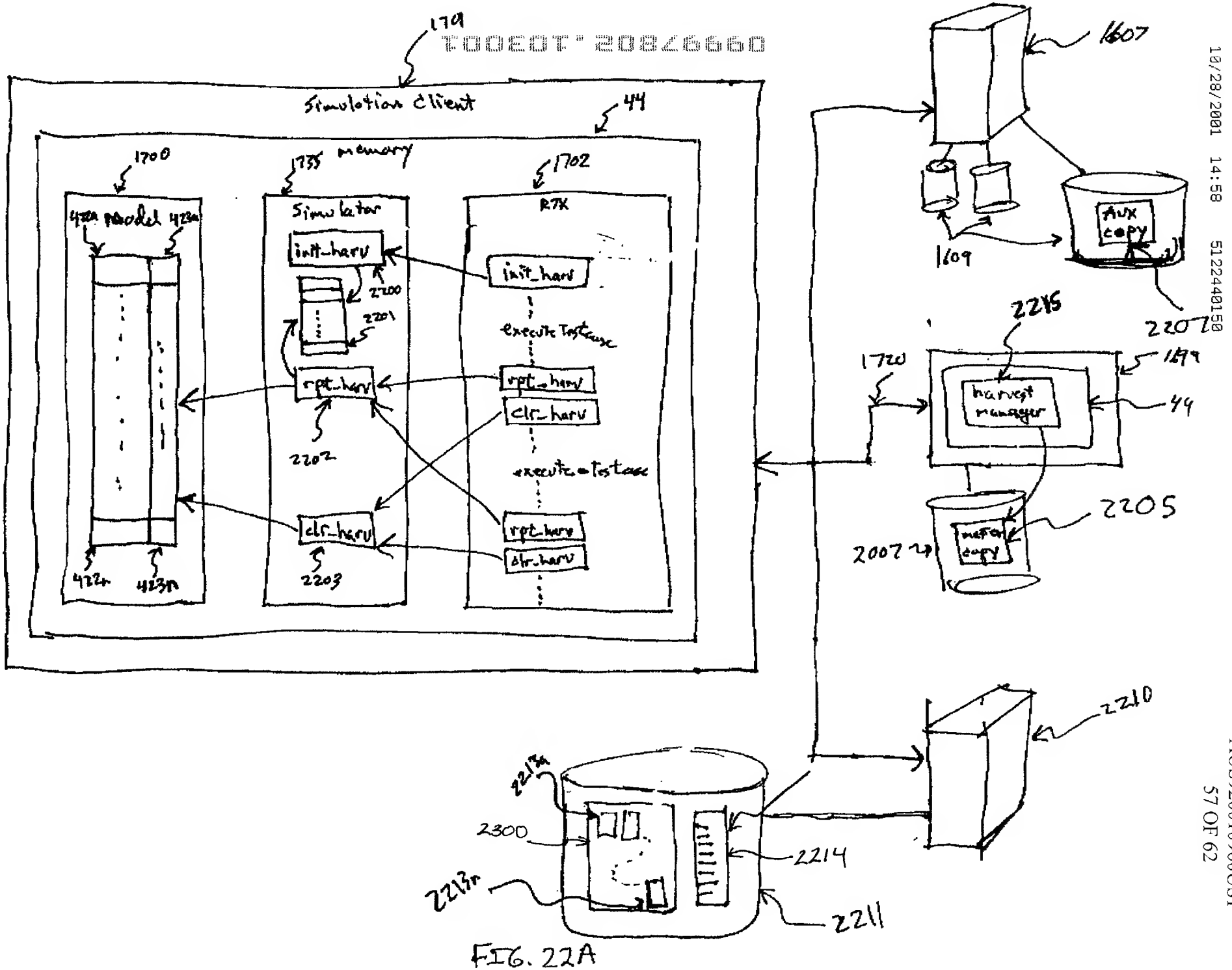

179
"TOOEOT" 20826660
Simulation Client
44
1700
memory
1735
1702
Simulator
RTX
model
init_harv
rpt_harv
clr_harv
execute Testcase
2200
2201
2202
2203
1607
1609
Aux copy
2207
2215
1720
1699
harvest manager
2205
2007
2210
2213a
2300
2214
2213n
2211

FIG. 22A

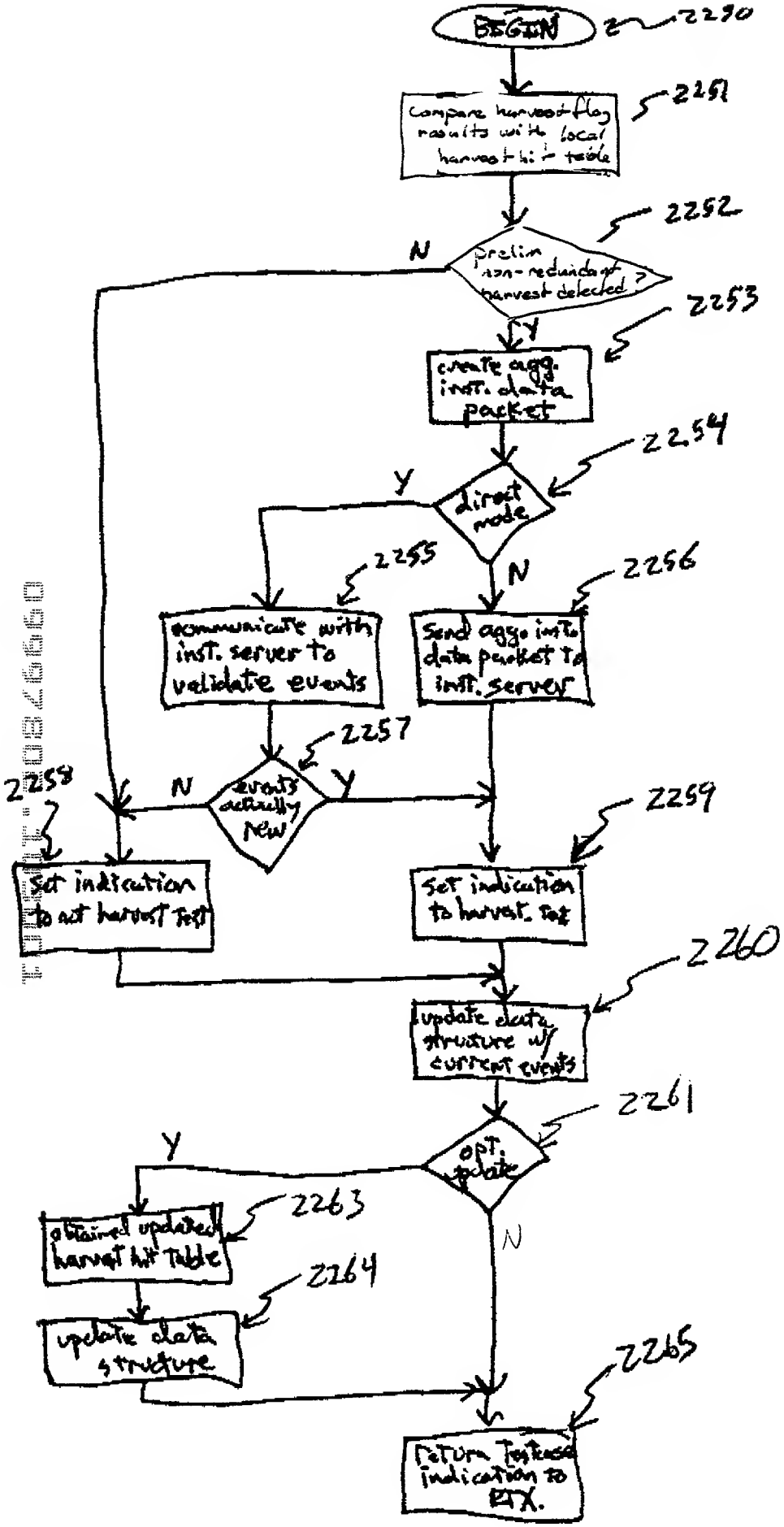

BEGIN
2250
Compare harvest flag results with local harvest hit table
2251
Prelim non-redundant harvest detected?
2252
N
Y
create aggr. instr. data packet
2253
direct mode
2254
Y
N
communicate with instr. server to validate events
2255
Send aggr instr. data packet to instr. server
2256
events actually new
2257
N
Y
2258
Set indication to set harvest bit
Set indication to harvest bit
2259
update data structure w/ current events
2260
opt. update
2261
Y
N
obtained updated harvest hit Table
2263
update data structure
2264
Return Trace indication to RTX.
2265

FIG. 22B

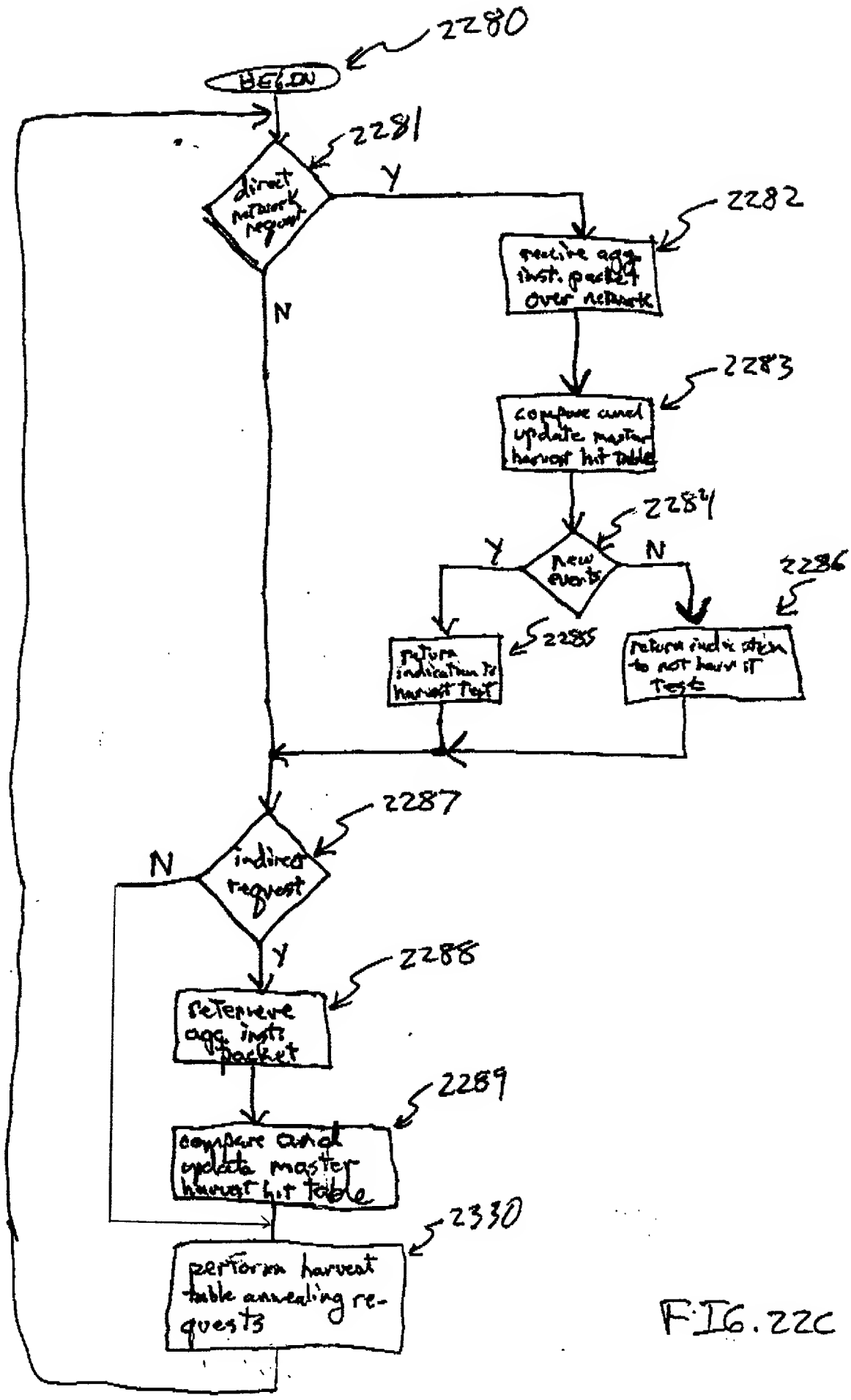


FIG. 22C

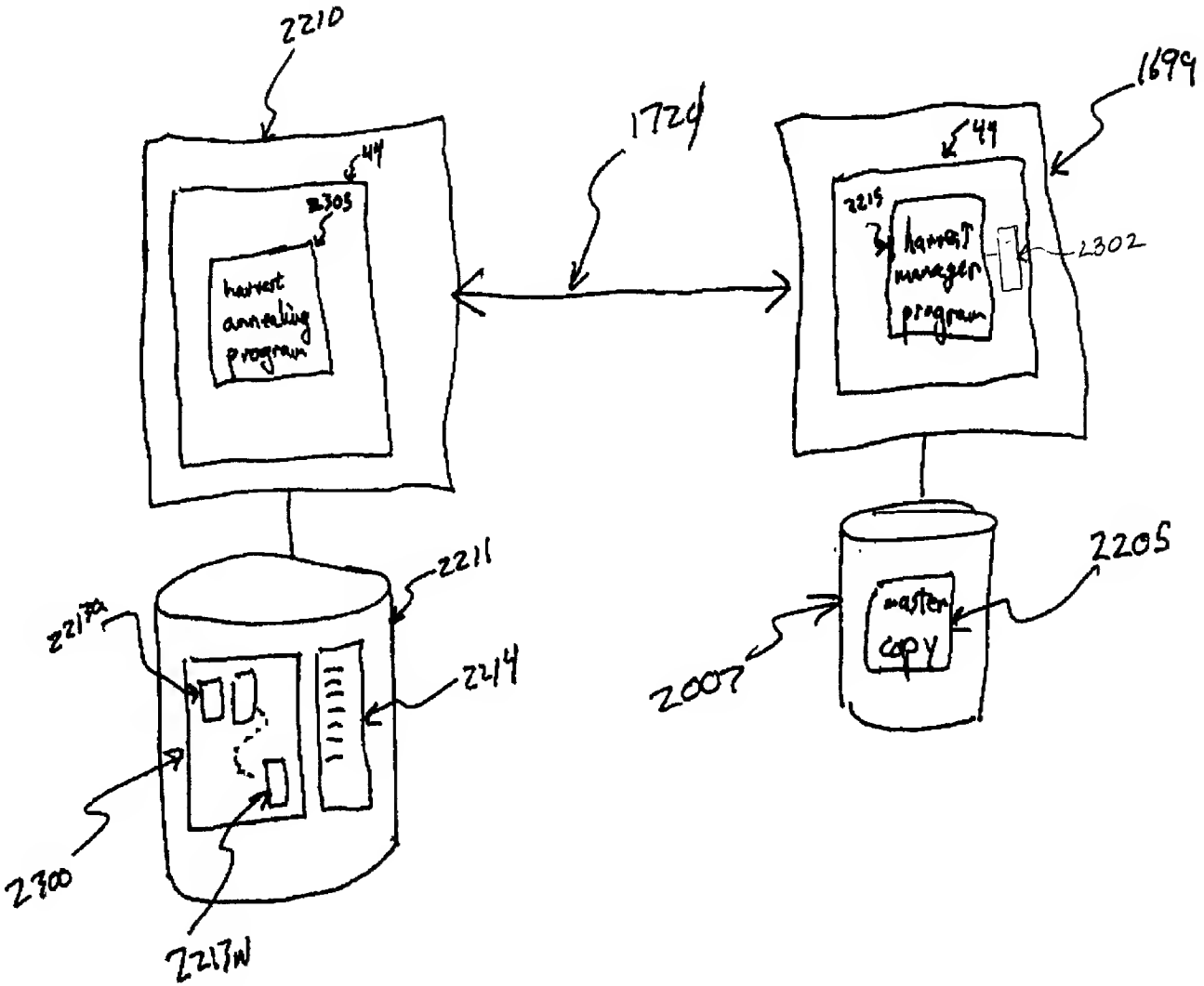


FIG. 23A

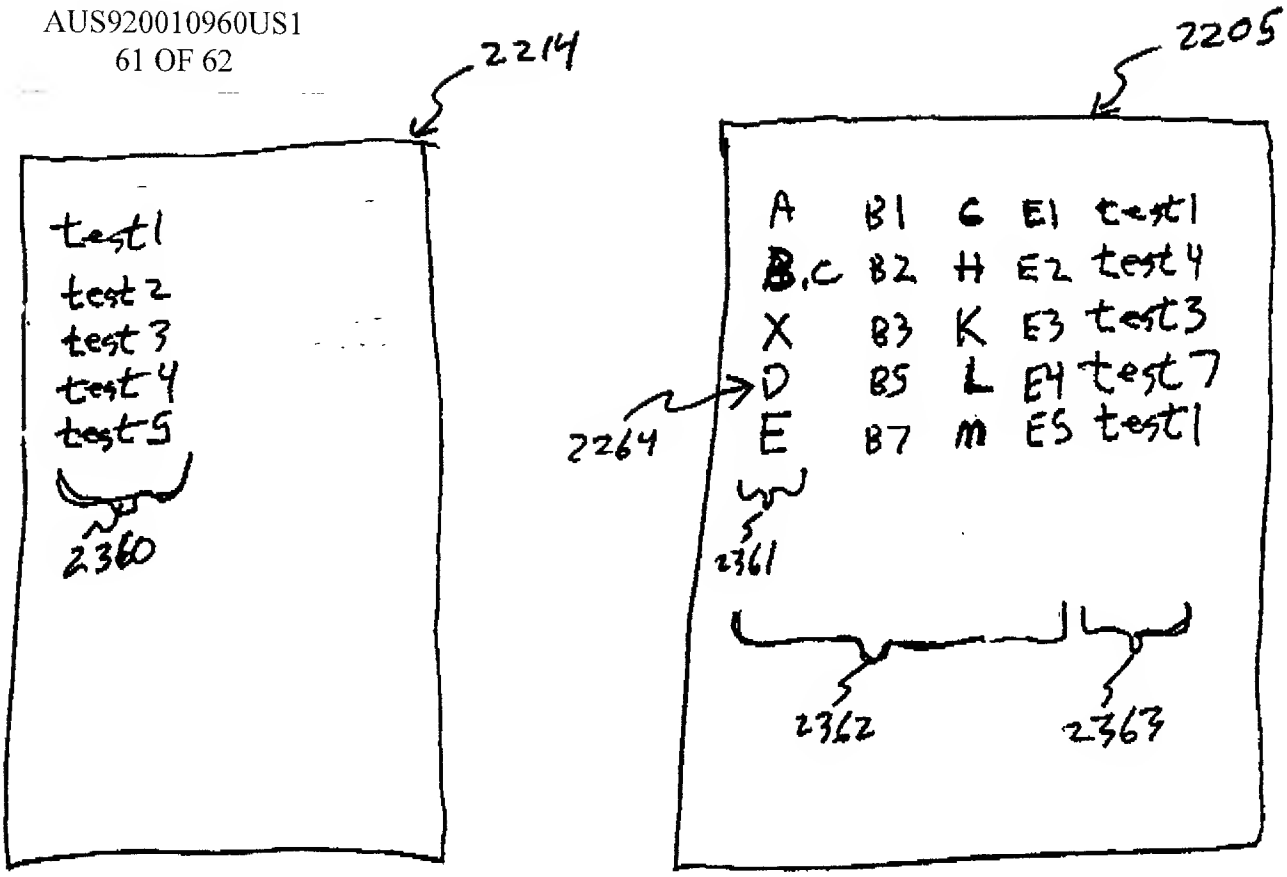


FIG. 23B

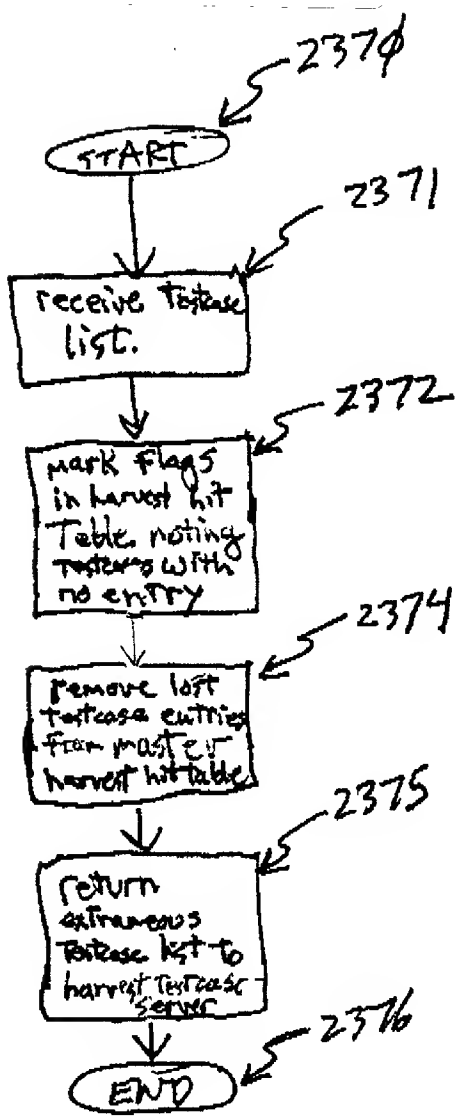


FIG. 23C